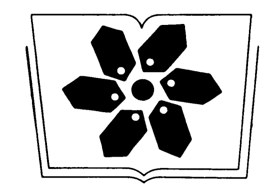


中国科学院科学出版基金资助出版

《数学机械化丛书》获国家基础研究发展规划项目“数学机械化与自动推理平台”与“数学机械化应用推广专项经费”资助

数学机械化丛书　　5

近世计算理论导引

——NP 难度问题的背景、前景及其求解算法研究

黄文奇　许如初　著

科 学 出 版 社

北　京

内 容 简 介

本书对迄今为止有关计算理论的实质性成果作了深刻、严格而又直观的论述,为计算机科学的实质性难题 NP 难度问题的实现求解提出了一条现实的高效的求解途径.它在透彻讲解图灵机的基础上,阐明了为什么会有计算机不可解的问题,会有计算机难解的问题;然后为当代实质性的计算机难解问题,即 NP 难度问题指明了得出高性能求解算法的现实途径——拟物、拟人途径;最后为设计算法与分析问题的复杂度提供了一个强有力的工具——有穷损害优先方法.

本书的内容经过不同组合可作为大学生、硕士生、博士生的教材,也可供有关的科技人员参考。

图书在版编目(CIP)数据

近世计算理论导引:NP 难度问题的背景、前景及其求解算法研究/黄文奇,许如初著.—北京:科学出版社,2004

(数学机械化丛书;5)

ISBN 7-03-012617-3

Ⅰ.近… Ⅱ.①黄…②许… Ⅲ.电子计算机-算法理论 Ⅳ.TP301.6

中国版本图书馆 CIP 数据核字(2003)第 125395 号

责任编辑:吕 虹/责任校对:柏连海

责任印制:钱玉芬/封面设计:黄华斌

科学出版社 出版

北京东黄城根北街16号

邮政编码:100717

http://www.sciencep.com

印刷

科学出版社发行 各地新华书店经销

*

2004 年 6 月第 一 版 开本:B5(720× 1000)

2006 年 6 月第二次印刷 印张:6 1/2

印数:3 001—5 000 字数:105 000

定价:20.00 元

(如有印装质量问题,我社负责调换〈新欣〉)

《数学机械化丛书》前言[①]

十六七世纪以来,人类历史上经历了一场史无前例的技术革命,出现了各种类型的机器,取代各种形式的体力劳动,使人类进入一个新时代.几百年后的今天,电子计算机已可以有条件地代替一部分特定的脑力劳动,因而人类面临另一场更宏伟的技术革命,处在又一个新时代的前夕.数学是一种典型的脑力劳动,它在这一场新的技术革命中,无疑地将扮演一个重要的角色.为了了解数学在当前这场革命中所扮演的角色,应对机器的作用,以及作为数学的脑力劳动的方式,进行一定的分析.

1. 什么是数学的机械化

不论是机器代替体力劳动,或是计算机代替某种脑力劳动,其所以成为可能,关键在于所需代替的劳动已经“机械化”,也就是说已实现了刻板化或规格化.正因为割麦、刈草、纺纱、织布的动作已经是机械化刻板化了的,因而可据此造出割麦机、刈草机、纺纱机、织布机来.也正因为加减乘除开方等运算这一类脑力劳动,几千年来就已经是机械地刻板地进行的,才有可能使得 17 世纪的法国数学家帕斯卡,利用齿轮传动造出了第一台机械计算机——加法机,并由莱布尼茨改进成为也能进行乘法的机器.数学问题的机械化,就要求在运算或证明过程中,每前进一步之后,都有一个确定的、必须选择的下一步,这样沿着一条有规律的、刻板的道路,一直达到结论.

在中小学数学的范围里,就有着不少已经机械化了的课题.除了四则、开方等运算外,解线性联立方程组就是一个很好的例子.在中学用的数学课本中,往往介绍解线性方程组的各种“消去法”,其求解过程是一个按一定程序进行的计算过程,也就是一种机械的、刻板的过程.根据这一过程编成程序,由电子计算机付诸实施,就可以不仅机器化而且达到自动化,在几分钟甚至几秒钟之内求出一个未知数多至上百个的线性方程组的解答来,这在手工计算几乎是不可能的.如

① 20 世纪七八十年代之交,我尝试用计算机证明几何定理取得成功,由此并提出了数学机械化的设想.先后在一些通俗报告与写作中,解释数学机械化的意义与前景,例如 1978 年发表于《自然辩证法通讯》的“数学机械化问题”以及 1980 年发表于《百科知识》的“数学的机械化”.二文都重载于 1995 年由山东教育出版社出版的《吴文俊论数学机械化》一书.经过 20 多年众多学者的努力,数学机械化在各个方面都取得了丰富多彩的成就,并已出版了多种专著,汇集成现在的数学机械化丛书.现据 1980 年的《百科知识》的“数学的机械化”一文,稍加修改并作增补,以代丛书前言.

果用手工计算,即使是解只有三四个未知数的方程组,也将是繁琐而令人厌烦的.现代化的国防、经济建设中,大量出现的例如网络一类的问题,往往可归结为求解很多未知数的线性方程组.这使得已经机械化了的线性方程组解法在四个现代化中起着一种重要作用.

即使是不专门研究数学的人们,也大都知道,数学的脑力劳动有两种主要形式:数值计算与定理证明(或许还应包括公式推导,但这终究是次要的).著名的数理逻辑学家美国洛克菲勒大学教授王浩先生在一篇有名的“向机械化数学前进”的文章中,曾列举了这两种数学脑力劳动的若干不同之点,我们可以简略而概括地把它们对比一下:

计　算	证　明
易	难
繁	简
刻板	灵活
枯燥	美妙

计算,如已经提到过的加、减、乘、除开方与解线性方程组,其所以虽繁而易,根本原因正在于它已经机械化.而证明的巧而难,是大家都深有体会的,其根本原因也正在于它并没有机械化.例如,我们在中学初等几何定理的证明中,就经常要依靠诸如直观、洞察、经验,以及其他一些模糊不清的原则,去寻找捷径.

2. 从证明的机械化到机器证明

一个值得提出的问题是:定理的证明是不是也能像计算那样机械化,因而把巧而难的证明,化为计算那样虽繁而易的劳动呢? 事实上,这一证明机械化的设想,并不始自今日,它早就为 17 世纪时的大哲学家、大思想家和大数学家笛卡儿和莱布尼茨所具有.只是直到 19 世纪末,希尔伯特(德国数学家,1862~ 1943)等创立并发展了数理逻辑以来,这一设想才有了明确的数学形式.又由于 20 世纪 40 年代电子计算机的出现,才使这一设想的实现有了现实可能性.

从 20 世纪二三十年代以来,数理逻辑学家们对于定理证明机械化的可能性,进行了大量的理论探讨,他们的结果大都是否定的.例如哥德尔(Gödel)等的一条著名定理就说,即使看来最简单的初等数论这一范围,它的定理证明的机械化也是不可能的.另一方面,1950 年波兰数学家塔斯基(Tarski)则证明了初等几何(以及初等代数)这一范围的定理证明,却是可以机械化的.只是塔斯基的结果近于例外,在初等几何及初等代数以外的大量结果都是反面的,即机械化是不可能的.1956 年以来美国开始了利用电子计算机做证明定理的尝试.1959 年王浩先生设计了一种机械化方法,用计算机证明了罗素等著的《数学原理》这一经典著作中的几百条定理,只用了 9 分钟,在数学与数理逻辑学界引起了轰动.一时

间,机器证明的前景似乎非常乐观.例如1958年时就有人曾经预测:在10年之内计算机将发现并证明一个重要的数学新定理.还有人认为,如果这样,则不仅许多著名哲学家与数学家,如皮亚诺、怀特海、罗素、希尔伯特以及图灵等人的梦想得以实现,而且计算机将成为科学的皇后,人类的主人!

然而,事情的发展却并不如预期那样美好.尽管在1976年,美国的哈肯等人,在高速计算机上用了1200小时的计算机时间,解决了数学家们100多年来所未能解决的一个著名难题——四色问题,因此而轰动一时,但是,这只能说明计算机作为定理证明的辅助工具有着巨大潜力,还不能认为这样的证明就是一种真正的机器证明.用王浩先生的说法,哈肯等关于四色定理的证明是一种使用计算机的特例机证,它只适用于四色这一特殊的定理,这与所谓基础机器证明之能适用于一类定理者有别.后者才真正体现了机械化定理证明,进而实现机器证明的实质.另一面,在真正的机械化证明方面,虽然塔斯基在理论上早已证明了初等几何的定理证明是能机械化的,还提出了据以造判定机也即是证明机的设想,但实际上他的机械化方法非常繁,繁到不可收拾,因而远远不是切实可行的.1976年,美国做了许多在计算机上证明定理的实验,在塔斯基的初等几何范围内,用计算机所能证明的只是一些近于同义反复的"儿戏式"的"定理".因此,有些专家曾经发出过这样悲观的论调:如果专依靠机器,则再过100年也未必能证明出多少有意义的新定理来.

3. 一条切实可行的道路

1976年冬,我们开始了定理证明机械化的研究.1977年春取得了初步成果,证明初等几何主要一类定理的证明可以机械化.在理论上说来,我们的结果已包括在塔斯基的定理之中.但与塔斯基的结果不同,我们的机械化方法是切实可行的,即使用手算,依据机械化的方法逐步进行,虽然繁复,也可以证明一些艰深的定理.

我们的方法主要分两步,第一步是引进坐标,然后把需证定理中的假设与终结部分都用坐标间的代数关系来表示.我们所考虑的定理局限于这些代数关系都是多项式等式关系的范围,例如平行、垂直、相交、距离等关系都是如此.这一步可以叫做几何的代数化.第二步是通过代表假设的多项式关系把终结多项式中的坐标逐个消去,如果消去的结果为零,即表明定理正确,否则再作进一步检查.这一步完全是代数的,即用多项式的消元法来验证.

上述两步都可以机械与刻板地进行.根据我们的机械化方法编成程序,以在计算机上实现机器证明,并无实质上的困难.事实上中国科学院数学研究所某些同志以及国外的王浩先生都曾在计算机上试行过.我们自己也曾在国产的长城203台式机上证明了像西摩松线那样不算简单的定理.1978年初我们又证明了

初等微分几何中主要的一类定理证明也可以机械化.而且这种机械化方法也是切实可行的,并据此用手算证明了不算简单的一些定理.

从我们的工作中可以看出,定理的机械化证明,往往极度繁复,与通常既简且妙的证明形成对照,这种以量的复杂来换取质的困难,正是利用计算机所需要的.

在电子计算机如此发展的今天,把我们的机械化方法在计算机上实现不仅不难,而且有一台微型的台式机也就够了.就像我们曾经使用过的长城 203,它的存数最多只能到 234 个 10 进位的 12 位数,就已能用以证明西摩松线那样的定理.随着超大规模集成电路与其他技术的出现与改进,微型机将愈来愈小型化而内存却愈来愈大,功能愈来愈多,自动化的程度也愈来愈高.进入 21 世纪以后,这一类方便的小型机器将为广大群众普遍使用.它们不仅将成为证明一些不很简单的定理的武器,而且还可用以发现并证明一些艰深的定理,而这种定理的发现与证明,在数学研究手工业式的过去,将是不可想象的.这里我们应该着重指出,我们并不鼓励以后人们将使用计算机来证明甚至发现一些有趣的几何定理.恰恰相反,我们希望人们不再从事这种虽然有趣却对数学甚至几何学本身也已意义不大的工作,而把自己从这种工作中解放出来,把自己的聪明才智与创造能力贯注到更有意义的脑力劳动上去.

还应该指出,目前我们所能证明的定理,局限于已经发现的机械化方法的范围,例如初等几何与初等微分几何之内.而如何超出与扩大这些机械化的范围,则是今后需要长期探索的理论性工作.

4. 历史的启示与中国古代数学

我们发现几何定理证明的机械化方法是在 1976 至 1977 年之间.约在两年之后,我们发现早在 1899 年出版的希尔伯特的经典名著《几何基础》中,就有着一条真正的正面的机械化定理:初等几何中只涉及从属于平行关系的定理证明可以机械化.当然,原来的叙述并不是以机械化的语言来表达的,也许就连希尔伯特本人也并没有对这一定理的机械化意义有明确的认识,自然更不见得有其他人提到过这一定理的机械化内容.希尔伯特是以公理化的典范而著称于世的,但我认为,该书更重要之处,是在于提供了一条从公理化出发,通过代数化以到达机械化的道路.自然,处于希尔伯特以及其后数学的一张纸一支笔的手工作业时代里,公理化的思想与方法得到足够的重视与充分的发展,而机械化的方向与意义受到数学家的忽视是完全可以理解的.但在电子计算机已日益普及,因而繁琐而重复的计算已成为不足道的现代,机械化的思想应比公理化思想受到更大重视,似乎是合乎实际的.

其次应该着重指出,我们在从事机械化定理证明工作获得成果之前,对塔斯

基的已有工作并无接触,更没有想到希尔伯特的《几何基础》会与机械化有任何关系.我们是在中国古代数学的启发之下提出问题并想出解决办法来的.

说起来道理也很简单:中国的古代数学基本上是一种机械化的数学.四则运算与开方的机械化算法由来已久.汉初完成的《九章算术》中,对开平、立方与解线性联立方程组的机械化过程,都有详细说明.宋代更发展到高次代数方程求数值解的机械化算法.

总之,各个数学领域都有定理证明的问题,并不限于初等几何或微分几何.这种定理证明肇始于古希腊的欧几里得传统,现已成为近代纯粹数学或核心数学的主流.与之相异,中国的古代学者重视的是各种问题特别是来自实际要求的具体问题的解决.各种问题的已知数据与要求的数据之间,很自然地往往以多项式方程的形式出现.因之,多项式方程的求解问题,也就自然成为中国古代数学家研究的中心问题.从秦汉以来,所研究的方程由简到繁,不断有所前进,有所创新.到宋元时期,更出现了一个思想与方法的飞跃:天元术的创立.

“天元术”到元代朱世杰时又发展成四元术,所引入的天元、地元、人元、物元实际上相当于近代的未知元或未知数.将这些未知元作为通常的已知数那样加减乘除,就可得到与近代多项式与有理函数相当的概念与相应的表达形式与运算法则.一些几何性质与关系很容易转化成这种多项式或有理函数的形式及其关系.这使得过去依题意列方程这种无法可循需要高度技巧的工作从此变得轻而易举.朱世杰 1303 年的《四元玉鉴》又给出了解任意多至四个未知元的多项式方程组的方法.这里限于 4 个未知元只是由于所使用的计算工具(算筹和算板)的限制.实质上他解方程的思想路线与方法完全可以适用于任意多的未知元.

不问可知,在当时的具体条件下,朱世杰的方法有许多缺陷.首先,当时还没有复数的概念,因之朱世杰往往限于求出(正)实值.这无可厚非,甚至在 17 世纪笛卡儿的时代也还往往如此.但此外朱世杰在方法上也未臻完善.尽管如此,朱世杰的思想路线与方法步骤是完全正确的,我们在上世纪 70 年代之末,遵循朱世杰的思想与方法的基本实质,采用美国数学家里特(Ritt)在 1932,1950 年关于微分方程代数研究书中所提供的某些技术,得出了解任意复多项式方程组的一般算法,并给出了全部复数解的具体表达形式.此后又得出了实系数时求实解的方法,为重要的优化问题提供了一个具体的方法.

由于多种问题往往自然导致多项式方程组的求解,因而我们解方程的一般方法可被应用于形形色色的问题.这些问题可以来自数学自身,也可以来自其他自然科学或工程技术.在本丛书的第一本,吴文俊的《数学机械化》一书中,可以看到这些应用的实例.工程技术方面的应用,在本丛书中有高小山的《几何自动作图与智能 CAD》与陈发来和冯玉瑜等的《代数曲面造型》两本专著.上述解多项式方程组的一般方法已推广至微分方程的情形.许多应用以及相应论著正在

酝酿之中.

5. 未来的技术革命与时代的使命

宋元时代天元术与四元术的创造,把许多问题特别是几何问题转化成代数方程与方程组的求解问题.这一方法用于几何可称为几何的代数化.12 世纪的刘益将新法与“古法”比较,称“省功数倍”.这可以说是减轻脑力劳动使数学走上机械化道路的一项伟大的成就.

与天元术的创造相伴,宋元时代的数学又引进了相当于现代多项式的概念,建立了多项式的运算法则和消元法的有关代数工具,使几何代数化的方法得到了有系统的发展,俱见于宋元时代幸以保存至今的杨辉、李冶、朱世杰的许多著作之中.几何的代数化是解析几何的前身,这些创造使我国古代数学达到了又一个高峰.可以说,当时我国已到达了解析几何与微积分的大门,具备了创立这些数学关键领域的条件,但是各种原因使我们数学的雄伟步伐就在这些大门之前停顿下来.几百年的停顿,使我们这个古代的数学大国在近代变成了数学上的纯粹入超国家.然而,我国古代机械化与代数化的光辉思想和伟大成就是无法磨灭的.本人关于数学机械化的研究工作,就是在这些思想与成就启发之下的产物,它是我国自《九章算术》以迄宋元时期数学的直接继承.

恩格斯曾经指出,枪炮的出现消除了体力上的差别,使中世纪的骑士阶级从此销声匿迹,为欧洲从封建时代进入到资本主义时代准备了条件.近年有些计算机科学家指出,个人用计算机的出现,其冲击作用可与枪炮的出现相比.枪炮使人们在体力上难分强弱,而个人用计算机将使人们在智力上难分聪明愚鲁.又有人对数学的未来提出看法,认为计算机的出现,将使数学现在一张纸一支笔的方法,在历史的长河中,无异于石器时代的手工方法.今天的数学家们,不得不面对计算机的挑战,但是,也不必妄自菲薄.大量繁复的事情交给计算机去做了,人脑将仍然从事富有创造性的劳动.

我国在体力劳动的机械化革命中曾经掉队,以致造成现在的落后状态.在当前新的一场脑力劳动的机械化革命中,我们不能重蹈覆辙.数学是一种典型的脑力劳动,它的机械化有着许多其他类型脑力劳动所不及的有利条件.它的发扬与实现对我国的数学家是一种时代的使命.我国古代数学的光辉,鼓舞着我们为实现数学的机械化,在某种意义上也可以说是真正的现代化而勇往直前.

吴文俊

2002 年 6 月于北京

序　言

教学经验说明,严格的逻辑证明能使学生将事物把握得十分确切.但长期囿于严格逻辑证明的人往往会失去直觉的功能,进而失去创造性.他们只能检验出错误和虚伪的东西,而不善于发现和创造出真实、美好的东西.

另一方面,经验也说明,直觉能使学生将事物看得很自然很亲切,最终会导致对事物透彻的理解与把握.但同样地,只喜欢直观领悟而没有经过系统严格的逻辑证明的训练的人,最终有可能变成口才很好的事后诸葛亮.在微妙复杂的形势下,他们往往无力辨别事物的真伪,将一些似是而非的感想当成科学的事实,一旦碰到困难问题,就感到茫然,拿不出真实有用的解决办法.尤其是,他们不具有把一件事情从头到尾彻底做完的能力,在工作中不能真正解决具体问题.

更为重要的是,教师本人,有必要成为严格逻辑与生动直观这两方面的典范.在进行严格证明时,不能避重就轻,将容易的问题仔细证明而将困难的问题一带而过.对于困难的问题更应当正面对待,将它仔细地全面铺展开来,证得一清二楚,并且证明的过程中还要能将符号式子的含义形象地显示出来,防止证明的过程成为被动的验证符号式子搬家演变的合法性的过程.在进行直观解释时,要能把握事物的实质性的形象与来龙去脉,而不能只涉及看来生动却属表面的那些现象.尤其是要不时地考察从似是而非的直观中得出的错误结论,从而不断地校正自己的直观,使其变得更为真实深刻和准确.

经验说明,一个好的理论,一个好的定理,其最原始的创造性部分绝不可能从什么高级的体系经由单纯的逻辑推演而来,而只能是先朦朦胧胧地感觉出来或说是看出来,然后才寻求道理将它说清楚.因此教师教书时最重要的事情是要能向学生描述这个理论或这个定理是如何被想到的,同学生讨论能否很自然地就看出这个理论与定理来.否则教师治学再严谨,论述推理的逻辑再严格学生还是不懂,学生当时被迫于逻辑,承认了理论的正确性,甚至也建立了信仰,并且随之也正确地演算了大量的习题,但是事隔不久他们还是会遗忘,没有留下什么体会,没有形成自己的独立能力,遇到有关的新问题时仍然解决不了.这样的痛苦疲惫要经过很长的反复过程才有可能缓解,学生对事物的领悟水平和独立的思考能力才有可能稍有提高.更有甚者,对于有些深刻的理论,包括定理和算法,大多数学生虽然能照猫画虎地套用并且能得出正确的结果,但是毕其终生对于这些理论本身却没有达到真实的理解,没有领悟到要害,没有形成自己的体会.

许多大师们的教学经验说明,在一段时间内,学生学的东西太多,不但无益,

反而有害,那些东西对实质性的应该牢靠掌握的东西的理解不但没有帮助反而形成了干扰.特别是那些无穷无尽的细小技巧,永远也学不完,而且一旦陷入之后,学生往往会忘记事物的本质和主流,最后成为精巧的工匠和事务主义者.学生们在掌握数量并非十分庞大的最本质的理论和最基本的技巧之后应立即从事世界前沿的科学研究,在这种真实的研究工作中进行有选择性的深入细致的思考和试验,以提高自己的理论水平和实际工作能力.在这一过程中用功细心的学生自然会自己创造发明出许多细小的技巧并有能力自觉地学会过去在学校没有学过的相关的有用理论和技术.

在现代科学技术中有一个令人十分向往的领域,那就是人类思维劳动的机械化.虽然全人类对此问题已思考了数千年之久,但是直到19世纪末才产生了颇具规模的有关科学技术.然而,至20世纪70年代,这方面的科学技术已有了长足的发展和进步.由于微型电脑的出现,由于电脑在各种控制工程中的应用,特别是电脑在现代通信网络中的管理与控制的作用,计算机科学技术已深深地渗入到人类生活的各个方面.

本书的目的是希望对人类的思维劳动机械化的有关哲学从科学与技术的角度加以表述,以期望有关学生提高自己的科学素养并获得若干基本的实用技术.希望本书的表述能做到既严格又直观,尽量浅显易懂,内容则在保存本质的条件下尽量精简.作者估计,不管学生毕业后将来从事什么样的工作,这种素养和技术对他们都是有益有用的.

由任课教师作不同方式的讲授和发挥,选择本书的不同章节可以构成大学各类院系的教本或参考书.学生的对象可以是本科生也可以是硕士生或博士生.学生专业的门类可以是计算机科学技术、哲学、数学、自动控制、无线电通信、工商管理等,也可是任何其他的理科和工科专业.

本书第一章描述作为计算的数学模型的Turing机,包括它的严格数学定义与直观形象.第二章说明世上确实有许多事情是无法办到的,然后严格证明停机问题的不可解性及Gödel不完全性定理.第三章讲解NP完全理论,显示并论证在现实生活中频繁地出现的大量问题——NP难度问题对于数学家与计算机科学家来说是困难的,并指出这种困难的现象与根源.第四章介绍对于NP难度问题获得高性能求解算法的一个有效途径——拟物拟人.此章为本书的核心,它可供具有不同优势的学者单独阅读.特别地,可供稍具物理知识的数学家阅读,也可供对数学有点兴趣的物理学家阅读.在作者看来,计算机科学家也就是数学家,因为计算机科学事实上就是数学科学中的一个特殊门类,它是侧重于时间的数学,不像20世纪中叶以前的数学那样侧重于空间.第五章介绍一种有用的工具——有穷损害优先方法.这是一个来自于纯粹数学的方法,它对设计算法与研究计算复杂度的结构十分有用.然而由于行业的隔阂,当今世上计算机科学的专

家多不了解这一方法,而少数了解这一方法的数学家中也乏人参与计算机算法的设计.

本书的雏形曾于 1991 年在吴文俊先生主持、由中国国家自然科学基金委员会支持的南开数学研究所计算机数学年的系列活动中作为博士生课程“计算复杂性理论导引”的讲义由黄文奇作过系统的讲授.在 1991 年至今的 10 余年中余新国先生、宋恩民博士、金人超博士及作者黄文奇、许如初曾用此书的初稿为华中科技大学及其前身华中理工大学计算机科学技术领域的硕士生、博士生作过近 20 次的系统讲授.其间偶有外系、外校及科学院所的青年学子参与听课.

本书的部分章节曾被作者在国内外的一些大学与科学院所作为讲义讲演过,其中包括北京大学、清华大学、中国人民解放军长沙国防科技大学、北京航空航天大学、武汉大学、哈尔滨工业大学、重庆大学、兰州大学及郑州大学;包括中国科学院数学研究所,系统科学研究所,自动化研究所及软件研究所;包括香港大学,香港中文大学,香港城市大学及香港浸会大学;包括美国康乃尔大学,新加坡国立大学及法国比卡地·朱尔丝文尼大学.有听讲者反映本书对计算理论的最实质性部分都讲到了,但比已有的其他书籍通俗好懂,并且某些章节在当今还有其实用价值;缺点是对某些有用的理论与技术没有提及,有的提及了又不够深入细致.为了补充本书的不足,建议使用本书的教师和学生参考以下四本教科书和专著:

1　Martin D. Davis and Elaine J. Weyuker, Computability, Complexity, and Language (fundamentals of theoretical computer science), Academic Press, INC., 1983.[中译本:戴维斯与维俞克,可计算性,复杂性和语言(理论计算机科学基础),张立昂,陈进元,耿素云译,清华大学出版社,北京,1989.]

2　Robert I. Soare, Recursively Enumerable Sets and Degrees —— A Study of Computable Functions and Computably Generated Sets, Springer-Verlag, Berlin Heidelberg New York London Paris Tokyo, 1987

3　Harry R. Lewis and Christos H. Papadimitriou, Elements of the Theory of Computation, Prentice-Hall, Inc. and 清华大学出版社,北京,1999.[中译本:勒维斯与帕帕蒂米特里欧,计算理论基础,张立昂,刘田译,清华大学出版社,北京,2000.]

4　张健,逻辑公式的可满足性判定——方法、工具及应用,科学出版社,北京,2000.

本书许多根本的思想、感觉和技术都是来源于作者的老师与学长,没有他们几十年来的熏陶、教诲、指导与帮助,作者对有关的世界不可能达到目前的认识.他们是周培源、李学英、朱九思、程民德、余家荣、吴文俊、王浩(Hao Wang)、王世强、黄敦、陆钟万、陈廷槐、陈耀松、涅罗德(Anil Nerode)、杨东屏、陶仁骥、董韫

美、陈火旺、石赫、李慧陵、黄泽权(C.K.Wong)、杨乐、黄且圆、索阿(Rorbert I. Soare)、李未、葛可一(K.I.Ko)、堵丁柱、唐守文.

作者进入SAT问题非完整求解算法领域的研究是源于李未的启发与激励,在此研究的初期阶段,研究工作是在他的领导之下并与他合作进行的.

对于原苏联的物理学家Л.Д.Ландау,作者虽无缘面见,但已被他的著作所深深感动.相关的读者会察觉到本书的正文是明显地受到了他的影响,只不过有些表述远不及他的美丽、清晰和简单.

作者的青年同事李初民(Chumin Li)、柳渝(Yu Li)和吴有亮(Yu-Liang W),依他们的学识见地对作者的研究工作给予了大的激励并开阔了作者的眼界.青年同事詹叔浩,依其才能与勤奋,同作者一道在拟物算法的最初形成阶段完成了因试探而动荡不定的艰苦的计算工作.

另外,作者过去有幸带了许多天赋很好的学生,其中有的直接参与了作者所进行的科学研究的主流工作中具体问题的思考和计算.本书某些部分事实上隐含着他们的智慧与劳动.他们是余向东、陈亮、金人超、朱虹、许向阳、陈广华、赵孝武、陈卫东、张德富、康雁.

一个多少带有偶然性的喜剧性事件是,以本书第四章的哲学和技术为基础的Solar算法在“第三届SAT问题快速算法国际竞赛”上获金奖(第1名).这届竞赛于1996年3月15日至17日在中国北京举行.第二届与第一届是分别于1993年,1992年在美国与德国举行.预计类似的科学测量性质的国际竞赛今后还会不断地在世界大都市举行.

自1985年至今,18年来,作者的研究工作持续地得到了中国政府和研究机构的资助,它们是:国家自然科学基金,国家高技术研究发展计划(863),国家重点基础研究发展规划(973),中国高等学校博士学科点专项科研基金,中国科学院软件研究所计算机科学开放实验室课题基金.

作者热诚欢迎海内外专家学者及青年学生指出书中的错误与不足之处,提出改正或改进的建议,也欢迎他们就有关问题的看法作更深入细致的思辨.作者的通信地址是:

武汉市洪山区华中科技大学计算机学院(邮政编码:430074)

黄文奇　许如初
2003年10月于武汉

目　录

第一章　计算的数学模型——Turing 机 …… 1
§1. Turing 机的定义及其直观形象 …… 2
§2. Turing 机所计算的函数和所接受的语言,计算复杂度 …… 6
§3. Church-Turing 论题 …… 7
§4. Turing 机的编码 …… 8
第二章　不可计算性 …… 11
§1. 胜弈机之不存在性 …… 12
§2. 不可计算函数的存在性 …… 12
§3. 停机问题的不可解性 …… 14
§4. Turing 机停机问题之 Turing 机不可解性 …… 16
§5. Gödel 不完备性定理 …… 16
第三章　NP 完全理论 …… 18
§1. 增长速度 …… 19
§2. P 和 NP …… 22
§3. Cook 定理 …… 36
§4. 另外几个 NP 完全问题 …… 40
第四章　现实生活中的 NP 难度问题及其现实处理方法——处理 NP 难度问题的拟物拟人途径 …… 47
§1. 求解 Packing 问题的拟物方法 …… 50
§2. 求解覆盖(Covering)问题的拟物方法 …… 53
§3. 求解 SAT 问题的拟物方法 …… 54
§4. 求解不等圆 Packing 问题的拟物拟人方法 …… 57
§5. 求解 SAT 问题的拟物拟人方法 …… 62
§6. 求解不等圆 Packing 问题的纯粹拟人方法 …… 68
第五章　设计算法与研究计算复杂度的结构的一个工具——有穷损害优先方法 …… 71
§1. 递归论中的几个基本概念 …… 73
§2. 单纯集的存在性的构造性证明 …… 75
§3. 对有穷损害优先方法的几点评注 …… 78
参考文献 …… 79

第一章　计算的数学模型——Turing 机

什么是计算机？它是帮助人们进行脑力劳动的工具.因为当今的计算机皆以“电”作为物质载体,所以有许多人称其为电脑,这也是颇有道理的.至于说“电脑这个词汇即暗示着计算机可以完全取代人脑进行思维”,那是一种误会,不难澄清,也不会导致太实际的消极的影响.

自古以来,人体的手指、脚趾,以及绳结、算筹、算盘、手摇计算机、电动计算机等都是处于某种发展阶段的计算机.但是只有在物质硬件上将“判断”和“选择执行”串通起来,使得视格局情况而进行有限的循环计算成为现实以后,计算机才在真正的“自动化”上迈出了第一步.这就是现代电子计算机的产生,以后发展的路还很长很长.

为了能够精确、系统地对计算和计算机进行研究,需要以人类高度文明的产物——严格精致且十分明白的数学作为工具,这就产生了一个数学模型的问题,即研究什么是计算,什么是计算机,如何能十分明白地将它说清楚.为了有一个明白的表述,我们不得不在真实性和永恒性上做一些牺牲,即我们的概念只能覆盖人类过去数千年的认识,只能涉及到有关事物在今后数百年至多几万年以内的发展,而决不能预料多少万万年以后和离银河系十分遥远的地方发生的事情.有趣的是,虽然人类关于“计算”的概念是发源于纯粹数学的“整数”,但是,“计算”本身却不是一个纯粹数学的概念,不能从纯粹数学得到.从根本上讲它与物质材料及其运动有关.

在人类使用、观察并制造了许许多多的计算机之后,在近代电子计算机诞生的前夜,英国人 Alan Turing(1912~ 1954)为计算机和计算定义了它们的数学模型——Turing 机.或者说 Turing 设计出了能够作为计算和计算机的统一的数学模型的机器——Turing 机.

Turing 机有两大特点.第一,极为简单明白.它十分好懂,十分亲近于人.学过一点算术的中等程度的小学高年级学生是完全能理解的.Turing 机已经明白和确切到了纯粹数学的程度,它已完全成了一个数学的结构,可以用确切的数学概念来对它进行描述和深入系统的研究.Turing 机的第二个特点是它的包容性.Turing 机,实际上不是一台机器,而是一类机器,是可数无穷台具体的机器.

自 20 世纪初至今严肃的数学家的研究证明,从可计算性的角度来看,即不计时间、空间开销的具体大小,则对于世上任何一件事,只要有一台计算机能做到,则必有一台 Turing 机,它也能做到.这就是所谓 Church–Turing 论题.

几乎与 Turing 同时还有好几种计算机被人想象出来,并且它们都与 Turing 机等价.但是这许多机器中以 Turing 机为最简单、最自然、最美丽.于是其他的机器作为计算的数学模型就逐渐被人们淡忘了.

本章将清楚描述 Turing 机以及 Turing 机的行为过程——计算,包括有关的基本现象与基本性质,以利于掌握计算的科学实质并体会其亲切的直观形象.

Church-Turing 论题是说从不计开销的具体大小的可计算性的角度讲,Turing 机是世上一切计算机中之功能最强者.因为涉及非数学的物质问题,它是永远不能在数学上被严格证明出来的.

不过即使有一天 Church-Turing 论题被发现是错误的,例如发现了某种“计算功能”更强的物质,从而导致更强功能的计算机的存在,那么我们今天的 Church-Turing 论题仍然是“真实的”,有意义的,因为它帮助我们讨论清楚了一种在将来看来是较为简单的“计算”的现象.将来的高功能的计算是今天我们所认识的计算的一种推广.作为局部真理,我们今天的计算理论仍然是真实有用的.这种关系就好像牛顿力学之相对于相对论力学.

值得一提的是,今天人们谈到的“量子计算机”,“并行机”,“蛋白质计算机”,在可以想象的真实性范围内还没有形成对于 Turing 机的突破.

§ 1.　Turing 机的定义及其直观形象

Turing 机在物质上由一条被划分为空白方格的左右无穷的条带和一根指针组成.机器有其由有穷个字母构成的字母表$\{s_1, \cdots, s_n\}$,有其有限个内部状态$\{q_1, \cdots, q_m\}$.

初始时刻,Turing 机带上的有穷个方格中分别被写上了字母表$\{s_1, \cdots, s_n\}$中的某一个符号;其他方格均为空白,记作 s_0.指针指着最左非 s_0 方格的左边一格,机器内部状态为 q_1.Turing 机往后的行为动作由一个由有穷条规则构成的集合所指挥.

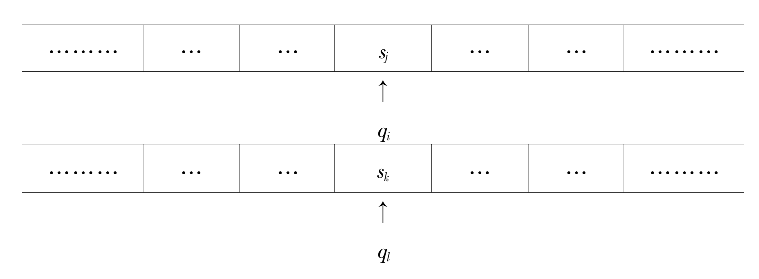
Turing 机的行为规则有如下三种类型,其外形为四重组.

① $q_i s_j s_k q_l$;

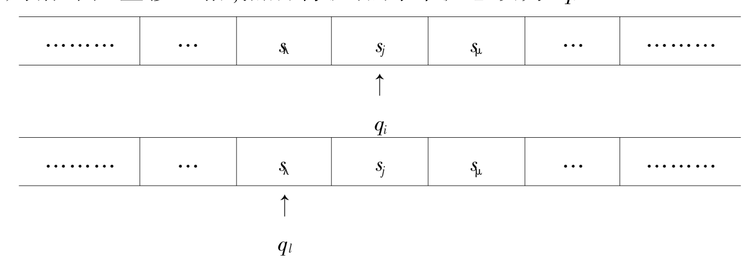
② $q_i s_j L q_l$;

③ $q_i s_j R q_l$;　　$0 \leqslant j, k \leqslant n,\ 1 \leqslant i, l \leqslant m$.

规则①的意义是:若 Turing 机当前的内部状态为 q_i,指针指着的方格上的符号是 s_j,则指针不移动,但将这格子上的符号 s_j 擦掉,接着写上符号 s_k,然后将机器的内部状态改为 q_l.



规则②的意义是:若 Turing 机当前的内部状态为 q_i,指针指着格子上的符号是 s_j,则指针往左移一格,然后将机器内部状态改为 q_l.



规则③的意义与规则②类似,只是指针的移动方向为右方.

以上规则的意义是,当时当地的内部状态和外部情况完全决定了 Turing 机的行为,而 Turing 机的每一步动作只是改变内部状态或当地的外部情况或局部地移动工作地点.

一台 Turing 机,在字母表$\{s_1,\cdots,s_n\}$及内部状态集$\{q_1,\cdots,q_m\}$给定后,完全由其行为规则集即四重组集所描述.

Turing 机如何确定当前的动作? 办法是想清自己当前的内部状态 q_i,再看清当地的外部状态,即指针现在所指的方格上的字符 s_j,然后再在行为规则集中查,看哪一条规则是适用的,即哪一条规则的打头二字为 $q_i s_j$.如查着了,就按那一条规则的指示进行动作,动作完成后再往下做;如查遍了还查不着就停机,即若没有有关的指示就不做动作.

显然,为了使 Turing 机在每一时刻都能确切地知道如何动作,而不致出现模棱两可的情形,指挥 Turing 机的动作的四重组集应该满足一个天然的约束条件,即四重组集中的任何两个四重组其打头的两个字不能完全相同.称这个条件为协调条件.

我们将字母表$\{s_1,\cdots,s_n\}$及内部状态表$\{q_1,\cdots,q_m\}$以及一个满足协调条件的四重组集联合在一起称为一台 Turing 机.

今观察如下实现函数 $f(x)=x+2$ 的计算的 Turing 机,其中空格被记作 B, s_1 被记作 1.

字母表:$\{1\}$,内部状态表:$\{q_1,q_2,q_3,q_4,q_5\}$,行为准则集为

q_1	B	R	q_2	…………进入计算
q_2	1	R	q_2	…………右移找尾
q_2	B	1	q_3	…………找到尾后进入加工态,先加 1
q_3	1	R	q_3	…………右移找 B
q_3	B	1	q_4	…………再加 1,进入回头状态
q_4	1	L	q_4	…………找左头

对函数值 $f(3)$ 的计算过程如下:

$\cdots BB111\ BB\cdots$

↑

q_1

$\cdots BB111\ BB\cdots$

↑

q_2

$\cdots BB111\ BB\cdots$

↑

q_2

$\cdots BB111\ BB\cdots$

↑

q_2

$\cdots BB111\ BB\cdots$

↑

q_2

$\cdots BB1111\ BB\cdots$

↑

q_3

$\cdots BB1111\ BB\cdots$

↑

q_3

$\cdots BB11111\ BB\cdots$

↑

q_4

$\cdots BB11111\ BB\cdots$

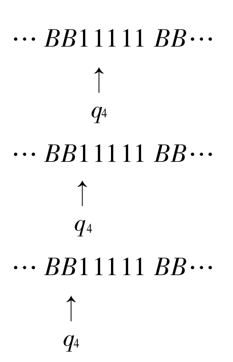
↑

q_4

$\cdots BB11111\ BB\cdots$

↑

q_4



因为四重组集中没有以 $q_4 B$ 打头的四重组,于是在此停机.带上有 5 个 1,表示函数值 $f(3)=5$.

再观察一台永不停机的 Turing 机.

字母表:{1},内部状态表:$\{q_1, q_2\}$,行为准则集为

$$\left(\begin{array}{cccc} q_1 & B & R & q_2 \\ q_1 & 1 & R & q_2 \\ q_2 & B & L & q_1 \\ q_2 & 1 & L & q_1 \end{array}\right) \begin{array}{l} \cdots\cdots\cdots\cdots q_1 \text{ 右移变 } q_2 \\ \\ \cdots\cdots\cdots\cdots q_2 \text{ 左移变 } q_1 \end{array}$$

即,遇内态为 q_1 时右移一格并将内态变为 q_2,遇内态为 q_2 时左移一格并将内态变为 q_1.

$B \times \times \times$
$\uparrow$
q_1

$B \times \times \times$
  $\uparrow$
  q_2

$B \times \times \times$
$\uparrow$
q_1

在任一给定时刻,Turing 机带上每一方格中为何符号,指针指在哪一个方格上,内部状态为第几个 q,这些都是确定的.我们将这些信息联合考虑在一起,称为 Turing 机的格局.如果不考虑 Turing 机的内部状态,则这些信息称为Turing机的带子上的格局——带格局.

对于给定的 Turing 机,当初始格局给定后,在四重组集的指挥之下,往后的发展是完全确定了的.或者是作有穷步动作之后停机,或者是永不停机.而不管是否最终会停机,每一步的动作都是完全明确的.

我们可以猜测,作为计算的数学模型的 Turing 机是 Turing 利用拟人的方法创造出来的.两端无穷的带可以被理解为外部客观世界.四重组集可以被理解为诸葛亮(奴隶主)给赵子龙(奴隶)的锦囊妙计,告诉赵子龙遇到什么情况就如何作.这个“情况”包括赵的自身状态 q_i 和赵在当时当地所面临的外部客观世界的状态 s_j.对外界状况 s_j 的了解是通过指针所模拟的奴隶的观察性能实现的.而擦写的动作及左右移的动作则是指针所模拟的奴隶的体力劳动,它们与奴隶的观察性能联合在一起实现了对奴隶主的命令的执行.而作为命令集的四重组集则是体现奴隶主的意志的改造客观世界的具体且完整的方法.

Turing 机就是这样的由奴隶主、奴隶与外部客观世界所构成的一个共同体.

§2.　Turing 机所计算的函数和所接受的语言,计算复杂度

2.1 Turing 机 M 所计算的 m 元($m \geqslant 1$)函数 f

设有字母表 $A = \{s_1, \cdots, s_n\}$, f 是从 $\underbrace{A^* \times A^* \times \cdots \times A^*}_{m个}$ 到 A^* 上的一个部分函数,即函数 f 在 A^{*m} 的某个子集上有定义,而在其余集上处处无定义,其中集合 A^* 的元素为字母表 $A = \{s_1, \cdots, s_n\}$ 上的字, $\underbrace{A^* \times A^* \times \cdots \times A^*}_{m个}$ 的元素为由 A^* 中的 m 个字拼接成的字.

设 Turing 机 M 的内部状态表为 $\{q_1, \cdots, q_v\}$,字母表为 $\{s_1, \cdots, s_n, \cdots, s_N\}$,说机器 M 实现了对函数 f 的计算是指如下意思:

若以 $\underset{\substack{\uparrow \\ q_1}}{s_0}\ x_1\ s_0\ x_2\ s_0 \cdots s_0\ x_m$ (其中 $x_i \in A^*$)的格局开始则最终 M 会停机当且仅当 $f(x_1, \cdots, x_m)$ 有定义,且在停机时的构形

$$\cdots\ s_0 \times \times \times \underset{\substack{\uparrow \\ q_i}}{\times} \times \cdots \times \times \times\ s_0 \cdots$$

中忽略掉所有的字符 $s_0, s_{n+1}, \cdots, s_N$ 后剩下的字符串即是字 $f(x_1, \cdots, x_m)$.

在这种情形,我们也称 f 为 Turing 机 M 所计算的 m 元函数,或称 Turing 机 M 计算了 m 元函数 f.

如果 M 计算了 m 元函数 f 且又满足以下两个附加条件,则称 M 严格地计算了 m 元函数 f:

1. $N = n$,即 Turing 机 M 没有使用 $\{s_0\} \cup \{s_1, \cdots, s_n\}$ 以外的符号;

2. M 停机时,带上构形如

$$\cdots\ s_0\ \underset{\uparrow}{s_0}\ y\ s_0\ s_0\cdots$$

其中字符串 y 不包含空白 s_0,即 $y\in A^*$.

已经证明,任一给定的 Turing 机计算的函数,一定能被某一 Turing 机严格地计算.

2.2 Turing 机 M 所接受的语言

Turing 机 M 所接受的语言 L 是 M 的字符表上的某些字(有穷字符串)所构成的集合.

设 Turing 机 M 的内部状态为$\{q_1,\cdots,q_m\}$,字母表为 $A=\{s_1,\cdots,s_n\}$,对于某个字 $u\in A^*$,如果以格局 $\underset{\underset{q_1}{\uparrow}}{s_0}\ u$ 开始 M 最终会停机,就称 M 接受字 u,M 所接受的语言 L 是 M 所接受的 A^* 中的字的全体所构成的集合.

于是,对于任一 $u\in A^*$,若 $u\in L$,则以格局 $\underset{\underset{q_1}{\uparrow}}{s_0}\ u$ 开始 M 最终会停机.若 $u\notin L$,则以格局 $\underset{\underset{q_1}{\uparrow}}{s_0}\ u$ 开始 M 会永不停机.

2.3 计算复杂度

在 2.1 中,设函数 f 为 Turing 机 M 所计算的 m 元函数,设 $f(x_1,\cdots,x_m)$ 有定义,于是当以格局 $\underset{\underset{q_1}{\uparrow}}{s_0}\ x_1\ s_0\ x_2\ s_0\cdots s_0\ x_m$ 开始时 M 最终会停机,我们将从开机始到停机止这整个过程中所执行过的四重组的次数简称为计算的步数,也称作 M 计算函数值 $f(x_1,\cdots,x_m)$的时间复杂度,将计算过程中带上最左一个曾经不为 s_0 的格子和最右一个曾经不为 s_0 的格子看成是带上"工作部分"的两端,这个"工作部分"所拥有的格子数称为机器 M 计算函数值 $f(x_1,\cdots,x_m)$的空间复杂度.

时间和空间复杂度又叫做时空开销.

§3. Church-Turing 论题

设有任意给定的字母表 $A=\{s_1,\cdots,s_n\}$,$n\geq 1$,f 是从 A^{*m} 到 A^* 上的一个任给的全函数,$m\geq 1$.Church-Turing 论题是如下断言:

只要世上有一台机器能够实现对函数 f 的计算,则一定存在一台 Turing 机,它能实现对函数 f 的计算.

Church-Turing 论题的意思是说,从可计算性的眼光来看,在世上所可能有的无穷多种计算机中,存在着功能最强的计算机种类,并且 Turing 机就是一个具体的功能最强的计算机种类.

这里我们可作如下三点注解.

注 1　这里的“一台计算机”也可以是由一支笔,一张纸及一个严格地按某个数学家已经说清楚了的规则进行计算的计算员所构成的三位一体.这个计算员身上带有数学家给他的计算规则说明书,当然这个计算员也可以由这个数学家本人来充当.

注 2　后来的研究说明,Church-Turing 论题中的全函数可以减弱为部分函数.原因是功能最强的计算机种类的存在,不同的计算机种类之间可以互相模拟对方的动作.甲机若不停机当然模拟它的乙机也不停机.

注 3　由于天书(oracle)的不存在性无法证实,Church-Turing 论题是永远不可能被证实为正确的.但是 Church-Turing 论题尚未被保证永远不会被证明为错误的,即可能世上仍然存在着某种充分简单清楚的,不神秘的装置,用它能计算 Turing 机无法计算的函数.然而许多有经验的数学家相信,这种装置并不存在.迄今为止,任何个人发现的具体的直观上的“可计算函数”,都是能被 Turing 机所计算的.

§4.　Turing 机的编码

$$\left(\{s_1, s_2\}, \{q_1, q_2, q_3\}, \left\{ \begin{matrix} q_2 & s_0 & s_1 & q_3 \\ q_3 & s_2 & L & q_2 \\ q_1 & s_1 & s_0 & q_2 \end{matrix} \right\} \right) \quad \text{是一台 Turing 机,}$$

$$\left(\{s_1, s_2, s_3\}, \{q_1, q_2\}, \left\{ \begin{matrix} q_1 & s_1 & s_2 & q_1 \\ q_2 & s_0 & s_1 & q_1 \\ q_1 & s_1 & L & q_2 \\ q_2 & s_2 & R & q_1 \\ q_2 & s_3 & s_0 & q_2 \end{matrix} \right\} \right) \quad \text{也是一台 Turing 机.}$$

世上的 Turing 机有无穷多台,我们将全体 Turing 机作个具体的统一编码,即将所有的 Turing 机排成一个序列,使得每一台 Turing 机在这个序列中恰出现一次.

在这种编码方案制定好后,我们可以说第 0 台 Turing 机,第 1 台 Turing 机,

第 5391 台 Turing 机等等,其中每一台 Turing 机都可以被我们确切地写出来,而对于任一给定的 Turing 机我们也可以说出它是第几台 Turing 机,即确切地说出它的码子.

两台 Turing 机被称为相同,是指它们具有相同的字母表和内部状态表,并且两个四重组集作为以个别的四重组为元素的集来看是相同的.

例如,四重组集 $\begin{pmatrix} q_2 & s_0 & L & q_3 \\ q_1 & s_1 & s_2 & q_4 \end{pmatrix}$ 与四重组集 $\begin{pmatrix} q_1 & s_1 & s_2 & q_4 \\ q_2 & s_0 & L & q_3 \end{pmatrix}$ 是相同的,而四重组集 $\begin{pmatrix} q_2 & s_0 & L & q_3 \\ q_1 & s_1 & s_2 & q_4 \end{pmatrix}$ 与四重组集 $\begin{pmatrix} q_2 & s_0 & R & q_2 \\ q_1 & s_1 & s_2 & q_4 \end{pmatrix}$ 是不同的.

现在我们叙述一种最自然的编码方案.

字母和状态都涉及到自然数,我们规定对自然数用普通二进制写法.于是规定将 s_6 写成 $s\,110$,将 s_0 写成 $s0$,将 q_1 写成 $q1$,将 q_7 写成 $q\,111$等等.在这种约定下,将任意一台 Turing 机用自然地拉长的方式写成$\{s, q, 0, 1, R, L\}^*$上的一个字.如 Turing 机

$$\left[\{s_1, s_2\}, \{q_1, q_2, q_3\}, \begin{pmatrix} q_2 & s_0 & s_1 & q_3 \\ q_3 & s_2 & L & q_2 \\ q_1 & s_1 & s_0 & q_2 \end{pmatrix}\right] \tag{4.1}$$

被写成

$$s1\,s10\,q1\,q10\,q11\,q10\,s0\,s1\,q11\,q11\,s10\,Lq10\,q1\,s1\,s0\,q10 \tag{4.2}$$

显然,从 Turing 机的(4.2)型的表示式恢复出其(4.1)型的表示式是能行的且是惟一的.

对字母表 $U=\{s, q, 0, 1, R, L\}$ 上的全体有穷符号串的集合 U^* 有一个自然的编码,即长字在后,短字在前,对长度相同的字按字典码排序.将 U 中的字母 $s, q, 0, 1, R, L$ 分别记为 $l_1, l_2, l_3, l_4, l_5, l_6$ 后, U^* 的元素的编码序为

$$\varnothing; l_1, l_2, l_3, l_4, l_5, l_6; l_1 l_1, l_1 l_2, \cdots, l_6 l_6; l_1 l_1 l_1, \cdots \tag{4.3}$$

于是可以说 U^* 中的一个字的字码是多少,例如空字的字码是0,字 l_1 的字码是1,字 $l_1 l_2$ 的字码是 8,等等.

显然,有以下三项事实成立:

1) 任给定一台 Turing 机 T,据(4.2)及(4.3)式可算出其字码 $M(T)$来, $M(T)\in\{0, 1, \cdots\}$;

2) 若两 Turing 机不同,则其字码必不相同,即若 $T_1 \neq T_2$,则 $M(T_1) \neq$

$M(T_2)$;

3) 对任一自然数 ν, 是否有 Turing 机以它作字码, 即是否存在 T, 使 $M(T)=\nu$, 此问题是可判定的. 例如, 没有 Turing 机其(4.2)型的表示式为 ss, 即 $l_1 l_1$, 于是没有 Turing 机, 其字码为 7, 即对于自然数 7, 我们知道没有 Turing 机以它为字码.

这样, 我们作出了从全体 Turing 机集合到自然数集合的一个能行的、全的、一一的映射 M(图 1.1).

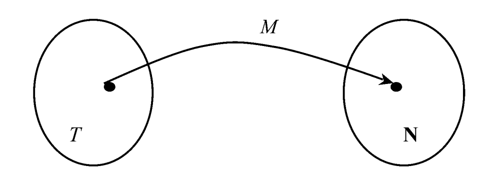


图 1.1

利用这个映射 M, 按如下方式即实现了对 Turing 机的编码:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	…
×	×	√	×	√	√	√	×	×	×	√	√	√	√	√	√	√	√	√	×	×	√	√	×	×	√	√	
		√		√	×	√				√	×	×	√	√	√	√	√	×			√	×			×	√	

1) 于第 1 行按顺序写出充分多的自然数;

2) 于第 2 行在每个自然数底下打钩或打叉, 打叉代表该自然数不是一 Turing机的字码, 打钩代表该自然数是某个 Turing 机的字码;

3) 于第 3 行由左至右在第 2 行的每个钩下打钩或打叉, 打钩代表第 2 行的该 Turing 机与其前面的所有 Turing 机都不相同; 打叉代表该 Turing 机与其前面的有些 Turing 机相同.

按照这种手续作下去, 与第三行的每一个钩号相应着一台 Turing 机, 第三行的钩号序列即是 Turing 机的序列, 并且世上任一 Turing 机在这个 Turing 机的序列中恰出现一次. Turing 机在这个序列中的位置号即是它的编码号.

显然, 任何两个不同的 Turing 机编码方案都是等价的, 即从一种码号到另一种码号之间的映射是从全体自然数到自身的一个可计算置换.

第二章　不可计算性

不可计算性是计算复杂性的一种最高级形态.

现实的不可计算性的存在就是计算机科学与技术的局限性.人们了解了这种不可能性之后就可以避免误入歧途,不将时间与精力投入那种做不到的事情中去,从而提高自己的思维效率.甚至在经受失败的时候,这种对不可计算性的认识还能帮助人达到自我的心理平衡,有利于身心的健康.

同时今天计算机科学所达到的对不可计算性的认识回答了一个有几千年历史的古老哲学问题,即思维机械化问题.这种认识,这种回答有利于人们科学素养的提高,有利于今后具体工作的开展.

在几千年以前,世上就有哲学家想象,能否设计一台机器,统一地解决所有的思维与辩论的问题.后来,这个思想被严肃、精密地考虑问题的科学家数学家笛卡儿、莱布尼兹、希尔伯特、塔尔斯基、哥德尔等发展到顶端.笛卡儿在引进坐标的概念后证实,数学中的某些思辨问题确实可以化归为计算,即可以用一台计算机来统一地加以解决.莱布尼兹思考了当时高性能计算机的设计问题.希尔伯特实际上正式提出了利用计算机彻底、系统地解决所有数学问题的计划.塔尔斯基利用精密的数学理论和技巧严格地证明了确实可以制造出一台计算机彻底系统地解决有关初等几何和初等代数的全部问题.这是人类有关计算机科学的第一次实质性的并有高显示度的光辉胜利.大自然与人类社会中有许多事比初等几何与初等代数要简单得多,因此人们显然可以制造出计算机以代替人来进行有关的思辨,并且速度与可靠度比人高.但不幸的是,哥德尔又几乎同时地证明了不存在着这样一种计算机,它能正确回答有关初等数论的全部问题.哥德尔的定理给了希尔伯特的纲领以彻底的否定.不过,希尔伯特的纲领仍然刺激了计算机科学的发展,使我们有了像塔尔斯基与哥德尔定理这样深刻的结果并获得了相应的理论方法和实用技术.特别值得一提的是,希尔伯特强调的公理化方法,在今日计算机科学技术的一个重要领域——软件工程中十分有价值,它能将计算机软件的生产过程变得非常地清晰明了,能极大提高软件产品的可靠性与可读性.

另外应该指出的是,现代中国以吴文俊为代表的中国人的初等几何判定算法,在大多数的情况之下比塔氏算法的性能高出很多,使判定时间从天文数字降低为几小时甚至几分钟或几秒钟.这就是说塔氏算法只能在天堂里供奉,而吴氏算法却可在人间施行.

中国古代有“以子之矛攻子之盾”的伟大思辨.利用这个思辨的精神,我们很容易证明,譬如关于象棋虽然存在着不败算法但不存在着恒胜的算法.因为如果你发明了这种算法,那么掌握这个算法的人与你对弈,你就不能取胜了.

沿着这个思路发展下去,如果能结合计算机可计算性问题的思考,利用简单的集合论的概念和技术,是有可能得出哥德尔不完全性定理的.可惜这一思路没有细致、严格、深入、系统地持续下去.

指挥计算机工作的算法,或者说程序,都只能是有穷字母表上的有穷长字母串.这就决定了世上全部不同程序的总数只有可数无穷多个($\aleph_0$).而以整数为定义域和值域的不同函数的总个数比可数无穷要多得多,其个数为不可数无穷($\aleph_0^{\aleph_0}$).

人们知道,“一把钥匙开一把锁”,7 把钥匙绝对开不了 8 把锁.因此,计算机绝对解决不了数论函数的描述问题,更解决不了有关数论函数的其他更复杂的问题.于是,计算机不可能全面彻底地一揽子解决所有的数论问题,这就很容易想象了.大自然与人类社会中的事,比数论现象更不知复杂多少倍,因此,也就不可能希望计算机能全面彻底地解决其中所有的问题.哥德尔不完全性定理及其证明就是将以上思考说严格了,说完整了.塔尔斯基与哥德尔的工作给笛卡儿-莱布尼兹-希尔伯特的思想路线划上了一个圆满的句号.用计算机科学技术的语言来说就是,计算机可作为人的一个很得力的助手,但它不能彻底地代替人的思考.

§1. 胜弈机之不存在性

假设存在一个计算机系统 A,下国际象棋能击败任何对手.于是存在一个被另一个人掌握的与 A 同样的计算机系统 $\neg A$,它下国际象棋能击败任何对手.

现在让 A 与 $\neg A$ 下国际象棋.有如下三种可能:① A 击败 $\neg A$;②下成平局;③ $\neg A$ 击败 A.情形①与前面的后一句话矛盾;情形②亦与前面的后一句话矛盾;情形③与前面的前一句话矛盾.即在任何情况下得出矛盾.

因此本节的第一句话为错,即根本不存在胜弈机.

§2. 不可计算函数的存在性

我们这里所说的函数是指从 N 到 N 的全函数,即映射 $f:\{0,1,2,\cdots\}\to\{0,1,2,\cdots\}$.

定理 1　不同程序的总个数不超过 N 的元素个数.

证　程序是一个有穷符号串,串中的每一个符号属于某有穷字母表

$\{a_1, \cdots, a_\nu\}$,其中常数 $\nu > 26$,有穷符号串共有可数个,即 $\aleph_0$ 个,排列如下:

$$\varnothing\ ; a_1, a_2, \cdots, a_\nu; a_1 a_1, \cdots, a_\nu a_\nu; a_1 a_1 a_1, \cdots, a_\nu a_\nu a_\nu; \cdots \tag{2.1}$$

在这个序列中有些串不代表程序,例如在用 C 语言编的程序中,若连续出现 *ggoottoo* 字样,则该串即不是合法程序.不过,对序列(2.1)中的任一串,我们是能判断它是否为合法程序的.总之,任一程序都是以上串序列中的某一串.于是,不同程序的个数 $\leqslant \aleph_0$.当然,从另一角度我们知道不同程序的个数 $\geqslant \aleph_0$,因为写上任意多句废话加在一个合法程序的某处后仍得合法程序. □

定理 2 数论函数有 $\aleph_0^{\aleph_0}$ 个.

证 一个函数 f 由它的函数表 $f(0), f(1), f(2), \cdots$ 确定.$f(0)$ 有 $\aleph_0$ 个可能方式取值;$f(0), f(1)$ 有 $\aleph_0^2$ 个可能方式取值;因此,$f(0), f(1), f(2), \cdots$,有 $\aleph_0^{\aleph_0}$ 个可能方式取值.即从 N 到 N 的函数共有 $\aleph_0^{\aleph_0}$ 个互不相同. □

定理 3 可计算函数总共恰有 $\aleph_0$ 个互不相同.

证 由定理 1 知程序至多只有 $\aleph_0$ 个互不相同,而每个程序至多只能计算一个函数,于是有程序计算的函数至多只有 $\aleph_0$ 个,即可计算函数至多只有 $\aleph_0$ 个.另一方面,显然,$0x, 1x, 2x, 3x, \cdots$ 皆为可计算函数,所以可计算函数恰有 $\aleph_0$ 个互不相同. □

定理 4 $2^{\aleph_0} > \aleph_0$.

证 $2^{\aleph_0}$ 代表不同的 01 无穷串的个数.如 000 …, 010101 …, 101001000100001…皆为 01 无穷串.显然,全体无穷 01 串的某个子集可与自然数集 N 一一对应,例如

$$\begin{array}{c} 1\ 0\ 0\ 0\ 0 \cdots \longleftrightarrow 0 \\ 0\ 1\ 0\ 0\ 0 \cdots \longleftrightarrow 1 \\ 0\ 0\ 1\ 0\ 0 \cdots \longleftrightarrow 2 \\ 0\ 0\ 0\ 1\ 0 \cdots \longleftrightarrow 3 \\ \cdots \end{array}$$

本定理是说,全体 01 无穷串太多,多得不可能与自然数集一一对应.

设能一一对应,则必有某种对应方式,设对应方式如下:

$$\begin{array}{c} a_{00}\ a_{01}\ a_{02}\ a_{03}\ a_{04} \cdots \longleftrightarrow 0 \\ a_{10}\ a_{11}\ a_{12}\ a_{13}\ a_{14} \cdots \longleftrightarrow 1 \\ a_{20}\ a_{21}\ a_{22}\ a_{23}\ a_{24} \cdots \longleftrightarrow 2 \end{array} \tag{2.2}$$

…

(2.2)式中左边的任一行代表一个 01 无穷串,其中每个 $a_{ij}=0$ 或 1;全体不同的 01 无穷串中的任一个串在(2.2)式的左端恰出现一次.

现构造一个 01 无穷串如下:$b_{00}\ b_{11}\ b_{22}\cdots$,其中

$$b_{ii}=\begin{cases}0, & 若\ a_{ii}=1,\\ 1, & 若\ a_{ii}=0,\end{cases}\qquad i=0,1,2,\cdots,$$

于是无穷 01 串 $b_{00}\ b_{11}\ b_{22}\cdots$在(2.2)式之左边不出现.矛盾. □

定理 5　存在着不可计算的函数.

证　定理 4 说明$\aleph_0^{\aleph_0}\geqslant 2^{\aleph_0}>\aleph_0$,即$\aleph_0^{\aleph_0}>\aleph_0$,即数论函数太多,程序能计算的函数太少,因此可知肯定有的数论函数无法用程序计算. □

即对任一计算机系统,纵然其内存与可运算时间要多大有多大,也都会有函数存在,使它无法计算.人类迄今为止的实践已经证明,在不考虑内存空间大小及计算时间长度的条件下,世上任意两个不太低级的计算机系统的能力都是一样的.于是知道,“存在着数论函数 $H(x)$,世上任何计算机系统都计算不了它.”

§ 3.　停机问题的不可解性

设 $T_0\ T_1\ T_2\ T_3\ T_4\cdots$为全体不同的合法程序的排序.对任一程序 T_i,当给了输入之后,启动,它可能会终究要停机,也可能永不停机.如果最终停机下来,则它有输出.对输入可编码为自然数,对输出也可编码为自然数.(2.1)式即是一种编码方案.因此,任一程序 T_i 即相当于一个函数φ_i,其定义域为$\{0,1,2,\cdots\}$,其值域为$\{\uparrow,0,1,2,\cdots\}$.例如 $T_i(8)=19$ 即表示用码为 8 的输入输进,开机一段时间后机器停止下来,输出为码与 19 相应的输出.$T_i(8)=\uparrow$即表示用码为 8 的输入输进,开机后永不停机.

因此,有对应

$$\begin{matrix}T_0 & T_1 & T_2 & T_3 & T_4 & \cdots\\ \varphi_0 & \varphi_1 & \varphi_2 & \varphi_3 & \varphi_4 & \cdots\end{matrix}$$

其中 φ_i 是 T_i 决定的一个从$\{0,1,2,\cdots\}$到$\{\uparrow,0,1,2,\cdots\}$上的映射(函数).

定义 1　说全体自然数的某子集合 A 为递归可枚举集(r.e. 集),是指它可以表示成某个φ_i 的定义域,即

$$(\exists\, i)[A=\{x\mid \varphi_i(x)\ 最终会停机\}]$$

或表示作$(\exists i)[A=\{x \mid \varphi_i(x)\downarrow\}]$

定义 2　说全体自然数的某子集合 A 为可计算集,是指存在着一个程序,对于任意给定的自然数 x, $x \in ?\ A$ 的问题是能由该程序确切判断的.

引理 1　可计算集为 r. e. 集.

证　设 A 为可计算集,定义可计算函数(即有程序计算的函数)

$$f(x)=\begin{cases}1, & 若\ x \in A; \\ \uparrow, & 若\ x \notin A.\end{cases}$$

于是 $A=\{x \mid f(x)\downarrow\}$.　□

引理 2　存在集 Θ,它不是 r. e. 集.

证　记集$\{x \mid \varphi_i(x)\downarrow\}$为 w_i,我们有对应:

T_0	T_1	T_2	T_3	T_4	$\cdots$
φ_0	φ_1	φ_2	φ_3	φ_4	$\cdots$
W_0	W_1	W_2	W_3	W_4	$\cdots$

只需找一集 Θ,它满足 $\Theta \neq w_i$, $i=0,1,2,\cdots$.因此可按如下对角线方法构造 Θ:
$i \in \Theta \Leftrightarrow i \notin w_i$, $i=0,1,2,\cdots$

即 i 构成 $\Theta \neq w_i$ 的证据(witness).

Θ 的定义式可简写成

$\Theta=\{i \mid i \notin w_i\}$,即$\Theta=\{i \mid T_i(i)$永不停机$\}$,或记为$\Theta=\{i \mid \varphi_i(i)\uparrow\}$.

由于 Θ 不等于任何一个 r. e. 集 w_i,所以它不是 r. e. 集.由引理 1 知 Θ 不是可计算集.　□

Θ 的余集通常记为 K,即 $K=\Theta=\{i \mid \varphi_i(i)\downarrow\}$.

引理 3　可计算集之余集为可计算集.

证　设 A 为可计算集,即有程序可回答 $x \in ?\ A$ 的问题,其中 x 为任给的自然数.于是按如下方法可回答 $x \in ?\ \neg A$ 的问题,其中 x 为任给的自然数:

若 $x \in A$,则回答 $x \notin \neg A$;

若 $x \notin A$,则回答 $x \in \neg A$.　□

定理 1　K 不是可计算集.

证　由引理 2 之后部知 Θ 不是可计算集. $K=\Theta$,于是由引理 3 知 K 不是可计算集.　□

定理 1 的科学意义为“停机问题不可解”,即不存在有如此强大功能的程序 G,它对任意给定的程序 P 和输入 x 可以判定程序 P 在给好输入 x 后进行计算是否会永不停机.

也就是说,严格的理论科学不是万能的,世间仍有些说得清楚的问题,它无法解决.

§4. Turing 机停机问题之 Turing 机不可解性

定理 1　任给自然数 x,问 $\varphi_x(x)$ 是否收敛,这个问题是不可能由某个 Turing机统一解决的,其中 φ_x 代表第一章§4 中所描写的第 x 号 Turing 机所计算的数论函数.

证　设问题可由某 Turing 机 T_θ 来统一解决.即有

$$\varphi_\theta = \begin{cases} s_1, & 若\ \varphi_x(x)\downarrow; \\ \phi, & 若\ \varphi_x(x)\uparrow. \end{cases}$$

定义函数

$$f(x) = \begin{cases} \uparrow, & \varphi_x(x)\downarrow; \\ s_1, & \varphi_x(x)\uparrow. \end{cases}$$

对此函数有

$$f(x) = \begin{cases} \uparrow, & T_\theta(x) = s_1, \\ s_1, & T_\theta(x) = \phi. \end{cases}$$

由 Turing 机的连接方法知有 Turing 机实现函数 $f(x)$的计算.设此 Turing 机的码号为 ν,于是有

$$\varphi_\nu(x) = \begin{cases} \uparrow, & \varphi_x(x)\downarrow; \\ s_1, & \varphi_x(x)\uparrow. \end{cases}$$

因此,$\varphi_\nu(\nu)\downarrow \Leftrightarrow \varphi_\nu(\nu)\uparrow$,矛盾.　□

§5. Gödel 不完备性定理

定理 1(Gödel)　初等数论的真命题中至少有一个不可能从 Peano 系统中得到证明.

证　设命题为假,即初等数论中的任一真命题皆可从 Peano 系统中得到证明.对于任一给定的自然数 x,$\varphi_x(x)\downarrow$ 与 $\varphi_x(x)\uparrow$ 这两个命题都是初等数论中的命题,且恰有一个为真,另一个为假.于是用枚举出 Peano 初等数论系统中的

全部真命题的方法可以判定问题“$\varphi_x(x)\downarrow$?”.

在用一台 Turing 机模拟以上枚举算法后便知,用某一台 Turing 机可以判定“$\varphi_x(x)\downarrow$?”问题,这与本章§4中的定理1矛盾. □

第三章　NP 完全理论

NP 难度问题是一大类问题,NP 完全问题是其中最基本的处于底层的最低级的一类问题.NP 完全问题约在 20 世纪 30 年代开始被人类认识其重要性.

NP 完全问题在科学哲学与现实生活中的重要价值在于它同时具有看来相反的两个性质,即通俗性和难解性.从各种纯粹的科学研究到不同阶层人民的日常现实生活,其中均频繁地涌现出大量各色各样的 NP 完全问题,它们既富理论意义又大有经济价值.这些问题外观上看起来都十分简单,中上等程度的初中三年级的学生都能在一个上午的时间里为其设计出精确且明白美丽的求解算法.但是对这些问题真实地求解起来却十分困难.原因是那些看起来精确美丽的求解算法,对于规模不是太小的问题运行起来动辄需要上万万年的时间.即使请来世上最聪明、造诣最高的数学家与计算机科学家,他们也设计不出精确而又不是太慢的求解算法来.这种情况是在 20 世纪 70 年代正式被澄清的.作这种澄清的道理即是所谓 NP 完全理论.从那时至今的研究进一步地验证了 NP 完全问题的这种"易"性中隐含着的"难"性,"简明性"中隐含着的"复杂性".这种情况进一步地使得数学中公理化学派的学者们觉得尴尬,20 世纪 30 年代的哥德尔定理只是说数学公理化方法对于某些数论问题是疲软无能的,而那些问题从某种意义上又可以说是矫揉造作地被人们为难倒公理化方法而特别不友好地制造出来的,它们特别艰深,在现实生活中其实没有多大意义.但是今天的情况就不一样了,NP 完全理论提出了在生活中频繁出现的各种数学问题,它们十分明了地呈现在人们面前,而数学公理化方法对它们从根本上讲却无能为力.

于是不少学者感觉到现有的数学公理化方法虽然有很多优点,是非常好的工具,但从根本上讲,它只能帮助人们整理思想,整理基本上已大致获得了的思维劳动的成果,而很难启发人们产生解决新问题的新智慧.因此当人们在探究新问题,特别是困难的新问题的时候,不能太依赖于公理化方法.还是要靠那个多少有点"神秘的",不太容易把握得住的"自身的心灵"以及这个心灵对外部客观世界的观察与感应.

NP 完全理论无疑在人类文明史上是一个很重要的发现,它同时具有重大的理论意义和实际价值.但目前学术界有学者对它有些误解,以为它很神秘,可能很能解决一些深奥的算法问题.其实这个理论很是明白,已没有什么疑点.另外,它也只是发现了一些问题在难度上是互相等价的,而没有指明任何一个具体问题究竟有多难,以及可能如何求解.所以,如果带有太实际的目的来研究 NP

完全理论,当研究清楚以后可能会有些失落感.NP 完全理论如果写成一篇论文,与其说是一篇优秀的“数学和计算机科学”论文,不如说是一篇优秀的“哲学”论文.它确实能提高计算机科学与数学方面各种学者与工程师的学术修养.美籍华人逻辑学家王浩(Hao Wang)在 NP 完全理论形成的过程中起了重要的作用.作为思想深刻又富实际经验的哲学家,王浩对命题逻辑中的合取范式有很深的直觉.Cook 在做学生的时候听过王浩的逻辑课,课上王浩曾具体讲授过他自己的心得,即合取范式的表达能力是如何之强,如何能模拟 Turing 机的计算行为.多少年后,Cook 发表了 NP 完全理论的核心定理——Cook 定理.此定理的证明过程实质上即是将“合取范式能模拟 Turing 机的计算行为”这句话说完整了、说严格了,Cook 的工作非常卓越.

由 Cook 定理自然地引出了一个问题,即 P 是否等于 NP.由于确定型的 Turing 机是不确定型的 Turing 机的特例,当然有 $P\subseteq NP$.现在的问题是 NP 是否属于 P,$NP\subseteq ?P$,这实质上也就是说对于 SAT 问题,是否存在着多项式时间的确定型的算法.这个问题被简称为 NP 问题.

王浩认为 NP 问题是当代计算机科学界、数学界和哲学界公共的重大实质性问题,同时具有重大的理论意义和实际意义.现在美国已有科学研究机构将此问题公开宣布为人类在 21 世纪最有意义的十大数学问题中的第一个问题.

另外,目前世界上还有许多工业、金融、商业、军事以及国家行政部门出重资支持对 NP 难度问题进行求解的研究.

对于 NP 问题的研究,笔者曾经浅尝,根据我们的经验,可以想象,虽然人类最终可能会取得完整的胜利,但是具体工作起来大致会异常地艰苦.

如果客观上 $P=NP$,要解决 NP 问题则要证明 $P=NP$.为此,人们必须为 SAT 问题的求解找一个多项式时间复杂度的确定型算法.这就需要非常高超精细的技术,最后事情会越扯越多,越扯越远以致难以收回.如果客观上 $P\neq NP$,要解决 NP 问题则要证明 $P\neq NP$.为此,人们必须证明,对于 SAT 问题的求解不存在多项式时间复杂度的确定型算法.这样,在证明的过程中人们多半会需要确立历史上未曾有过的新的数学公理.而这样的公理是极难确立的,因为第一它必须充分的简单,以致大多数头脑清醒的文盲都能接受,第二它又必须充分的复杂,事实上隐藏着很丰富的内涵,第三它还必须在广泛的范围内符合物质世界的客观事实.

但是不管怎样,人类总会对 NP 问题的认识逐渐深入,伴随着这种深入的过程会涌现出许多高雅的理论结果和有益的实用技术.

有志的科学青年不妨可考虑将研究此问题定为终生的奋斗目标.

§ 1. 增长速度

只考虑满足如下条件的数论函数 f:

$$\lim_{n\to\infty} f(n) = +\infty . \tag{1.1}$$

定义 1　$f(n) = O(g(n))$,其中 f, g 为从 N 到 N 的函数,是指存在自然数 C,使 $f(n) \leqslant Cg(n)$,对于充分大的 n 一致地成立.

若 $f(n) = O(g(n))$且 $g(n) = O(f(n))$,则称 f 与 g 有相同的速率.

对定义 1 可作如下直观形象的说明:

$f(n) \leqslant C_1 g(n) \leqslant C_1 C_2 f(n)$,说明 $C_1 C_2 \geqslant 1$ 且 $C_1 > 0$ 且 $C_2 > 0$, $f(n)/g(n) \leqslant C_1$, $g(n)/f(n) \leqslant C_2$.

正常数 $= \dfrac{1}{C_2} \leqslant f(n)/g(n) \leqslant C_1 =$ 正常数,于是称 $f(n)$, $g(n)$有相同的增长速度,即若不计正常数放大,二者速率相同(图 3.1).

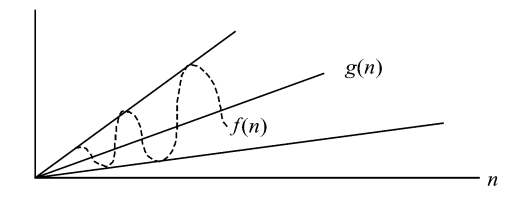


图 3.1

若 $f(n) = O(g(n))$, $g(n) \neq O(f(n))$,则称 $g(n)$比 $f(n)$增长快.

$g(n)$比 $f(n)$增长得快的直观意义是 $f = O(g)$,并且存在一个单调递增趋向 $+\infty$ 的自然数序列 $n_1, n_2, n_3, \cdots$,使得 $\lim\limits_{i\to+\infty} \dfrac{g(n_i)}{f(n_i)} = +\infty$.

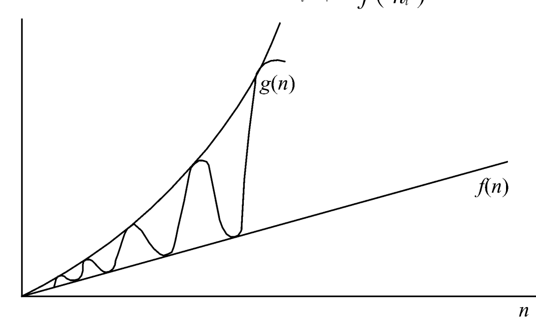


图 3.2

图 3.2 即是一个典型的例子.

例 1 $n^2 = O(3n^2 - 106n + 15)$.

因当 n 充分大后,

$$\frac{n^2}{3n^2 - 106n + 15} < 4,$$

于是 $n^2 \leqslant 4(3n^2 - 106n + 15)$ 对于充分大的 n 一致地成立.

例 2 $n^3 \neq O(3n^2 - 6n + 5)$.

若 $n^3 \leqslant C \cdot (3n^2 - 6n + 5)$ 对于充分大的 n 一致地成立,则 $C \geqslant \frac{n^3}{3n^2 - 6n + 5}$ 对于充分大的 n 一致地成立,于是常数 C 可大于任何常数,矛盾.

定理 1 设 f, g 为 $\mathrm{N} \to \mathrm{N}$ 的全函数, $\lim\limits_{n\to+\infty} f = +\infty$, $\lim\limits_{n\to+\infty} g = +\infty$,且 $\lim\limits_{n\to+\infty} \frac{f(n)}{g(n)} = \beta$,则

(i) 若 $\beta =$ 正实数,则 f, g 有相同的增长速率;

(ii) 若 $\beta = +\infty$,则 $f(n)$ 比 $g(n)$ 增长快;

(iii) 若 $\beta = 0$,则 $g(n)$ 比 $f(n)$ 增长快.

证 (i) 若 $\beta =$ 正实数,则 2β 及 $\frac{1}{2}\beta$ 皆为正实数.当 n 充分大时,有 $\frac{f(n)}{g(n)} \leqslant 2\beta$, $f(n) \leqslant 2\beta \cdot g(n)$,即 $f = O(g)$;又有 $\frac{f(n)}{g(n)} \geqslant \frac{1}{2}\beta$, $g(n) \leqslant \frac{2}{\beta} \cdot f(n)$,即 $g = O(f)$.

(ii) $\frac{f(n)}{g(n)} \to +\infty$, 于是 $\frac{f(n)}{g(n)} \geqslant 1$ 对于充分大的 n 一致地成立,于是 $g(n) \leqslant 1 \cdot f(n)$,对于充分大的 n 一致地成立,即 $g = O(f)$,另一方面若 $f = O(g)$,则存在正自然数 M,使得 $f(n) \leqslant M \cdot g(n)$ 对于充分大的 n 一致地成立,于是 $\lim\limits_{n\to+\infty} \frac{f(n)}{g(n)} \leqslant M \neq +\infty$,矛盾.

(iii) $\lim\limits_{n\to+\infty} \frac{f(n)}{g(n)} = 0$,则 $\lim\limits_{n\to+\infty} \frac{g(n)}{f(n)} = +\infty$,由(ii)可知 $g(n)$ 比 $f(n)$ 增长快. □

记 $a_r n^r + a_{r-1} n^{r-1} + \cdots + a_1 n + a_0 = P(n)$,其中 $a_{r-1}, \cdots, a_0$ 为整数, a_r 为正整数,称 r 为多项式 P 的次数.

定理 2 设 $P_1(n) = a'_{r'} n^{r'} + \cdots + a'_1 n^1 + a'_0$, $P_2(n) = a''_{r''} n^{r''} + \cdots + a''_1 n^1 + a''_0$, $a'_{r'}, a''_{r''} > 0$,则

(i) 若 $r' > r''$,则 $P_1(n)$ 比 $P_2(n)$ 增长速度快;

若 $r' < r''$,则 $P_2(n)$ 比 $P_1(n)$ 增长速度快;

(ii) 若 $r' = r''$,则 $P_1(n)$与 $P_2(n)$有相同的增长速率.

证 利用定理 1. □

定理 3 函数 k^n $(k>1)$比任何多项式的增长速度快.

证 只证特例,显示其直观形象.求证:$\frac{2^n}{n^5}\xrightarrow[n\to+\infty]{}+\infty$.

只需证 $\frac{2^{n/5}}{n}\xrightarrow[n\to+\infty]{}+\infty$,只需证 $\lim\limits_{n\to+\infty}\frac{2^{\frac{n}{5}}}{\frac{n}{5}}=+\infty$,只需证 $\lim\limits_{t\to+\infty}\frac{2^t}{t}=+\infty$,而对于大的 t 有

$$\frac{2^t}{t}=\frac{2\cdot2\cdot2\cdot2\cdot\cdots\cdot2\cdot2}{\frac{1}{1}\cdot\frac{2}{1}\cdot\frac{3}{2}\cdot\frac{4}{3}\cdot\cdots\cdot\frac{t-1}{t-2}\cdot\frac{t}{t-1}}$$

$$=\frac{2\cdot2\cdot2}{\frac{1}{1}\cdot\frac{2}{1}\cdot\frac{3}{2}}\cdot\frac{2\cdot\cdots\cdot2\cdot2}{\frac{4}{3}\cdot\cdots\cdot\frac{t-1}{t-2}\cdot\frac{t}{t-1}}$$

$$>\frac{2\cdot2\cdot2}{\frac{1}{1}\cdot\frac{2}{1}\cdot\frac{3}{2}}\cdot\left(\frac{2}{\frac{4}{3}}\right)^{t-3}=\text{正常数}\cdot(\text{大于 1 的数})^{t-3}$$

$$=\text{正常数}\cdot(\text{大于 1 的数})^t\xrightarrow[t\to+\infty]{}+\infty.$$

关键在于当$\frac{2^t}{t}$变成$\frac{2^{t+1}}{t+1}$时,分子放大了 2 倍,分母只放大了 $1+\varepsilon$倍,其中 ε 为小量$\frac{1}{t}$.因此整个分数放大了$\frac{2}{(1+\varepsilon)}$倍,因此每往前走一步几乎放大 2 倍,所以最终会趋于无穷. □

§2. P 和 NP

根据客观意义、主观美感及方便性等标准,定出问题的易算性标准如下:一个问题类,对它如果存在一个确定型的算法,伴随着存在一个多项式 $p(t)$,使得 $\forall x\in$ 此类问题,此算法都能在 $\leqslant p(|x|)$ 的时间内将问题 x 解出,其中 $|x|$ 为问题 x 的书写长度.

为使思想确切,我们借助 Turing 机模型.什么是“问题”,“问题类”,“算法”,“计算时间”? 这些概念都借助于 Turing 机说清.

Turing 机、Turing 机接收某个字以及 Turing 机接收语言 L,这些概念在第

一章已作介绍.

定义 1 字母表 $A=\{s_1,\cdots,s_n\}$ 上某些字的集合(语言) L 称为多项式可判定的,是指:

(i) 存在一个字母表为 $A=\{s_1,\cdots,s_n\}$ 的 Turing 机 T,它接收语言 L;

(ii) 伴随着 T,存在一个多项式 $p(n)$,使得对任一 $x\in L$ 都有

$$T\text{ 接收 }x\text{ 所经历的机器运行步数}\leqslant p(|x|).$$

此式左边称为计算时间;右边$|x|$表示字 x 中出现字母的次数,即 x 的字长.

对此定义可作如下解释.此机器 T 可用来解决如下问题:在 A^* 中任给出了一字 x,问 $x\in L$ 吗? 用此机器 T,只需在 x 输入后走 $p(|x|)$次,即经过时间长 $p(|x|)$,就能知道是否 $x\in L$.停机就说明 $x\in L$,尚未停机就说明 $x\notin L$,即存在机器 T,人们利用它能在多项式时间内解决 $x\in ?\ L$ 的问题,即语言 L 是多项式时间可判定的.

定义 2 当字母表 A 确定后,我们用 P 表示一切"多项式时间可判定的语言"所构成的类.

往往在说到 P 时,对字母表予以省略.

定义 3 设 A 是一个字母表,f 是 A^* 到 A^* 的一个全函数.如果存在计算 f 的 Turing 机 M 和多项式 $p(n)$,使得当输入为 $x\in A^*$ 时 M 的计算步数$\leqslant p(|x|)$,则称函数 f 是多项式时间可计算的.

注 1 说一个语言 L 是多项式时间可判定的,或说一个函数 f 是多项式时间可计算的,只需:存在一个多项式使得相应的 Turing 机 M 完成计算任务的计算步数$\leqslant p(|x|)$ 对充分大的$|x|$一致地成立,其中 x 为输入的字.

证 我们当然是假设了存在 M,它可被用来决定语言或计算函数 f.为了证明 M 实际上是在某多项式时间之内完成了任务,只需将多项式 $p(|x|)$改为新多项式 $p(|x|)+C$,其中 C 为不等式"…步数$\leqslant p(|x|)$ 对充分大的$|x|$一致地成立"中被忽略了的那有限个文字 x 分别被 M 机计算所花的时间的最大值. □

注 2 说一个语言 L 是多项式时间可决定的,或说一个函数 f 是多项式时间可计算的,可改定义为:存在相应的 Turing 机 M,它完成计算任务的步数 $\mathrm{step}(x)=O(|x|^r)$,此处 r 为某个正自然数.

证 此条件是充分的,因 $\mathrm{step}(x)=O(|x|^r)$意味着,除有限个 x 值外成立着 $\mathrm{step}(x)\leqslant C|x|^r$,其中 C 为某正常数.于是由注 1 知存在多项式 p,使 $\mathrm{step}(x)\leqslant p(|x|)$处处成立.

另一方面,此条件是必要的,因若处处成立着

$$\mathrm{step}(x)\leqslant a_0+a_1|x|^1+\cdots+a_r|x|^r,\quad a_r>0,$$

则 $\text{step}(x) = O(|x|^r)$. □

Cook-Karp 论题，任给字母表 $A = \{s_1, \cdots, s_n\}$，对 A^* 上的任一全函数 $f: A^* \xrightarrow{f} A^*$，若世间有某种(不管何种)装置能在多项式时间内计算函数 f，则必存在一台 Turing 机，它能在多项式时间内计算函数 f.

Cook-Karp 论题是说 Turing 机可以作为易计算之数学模型；而 Church-Turing 论题是说 Turing 机可以作为可计算性之数学模型. 两个论题联起来，说明 Turing 机作为计算装置具有实质性的代表意义. Turing 机在人类文明史上具有颇高的地位.

Cook，Karp 以及我国学者洪加威、唐守文都找到过一些证据来支持 Cook-Karp 论题. 有人认为洪-唐的证据比 Cook-Karp 的更多更有力. 王浩认为，目前的这些证据意义都不大.

定理 1　设 $L \in P$，f 为从 A^* 到 A^* 的一个多项式时间可计算函数，$Q = \{x \in A^* \mid f(x) \in L\}$，则 $Q \in P$(图 3.3).

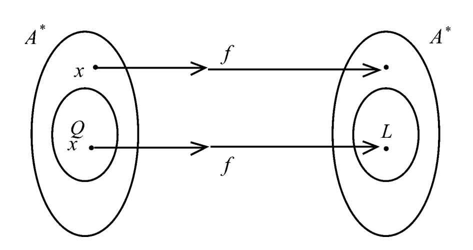


图 3.3

证　关键是 $Q = \{A^*$ 中的 f 像 $\in L$ 的点$\}$，因此 $(\forall x)_{\in A^*}[x \in Q \Leftrightarrow f(x) \in L]$. 因此，为回答 $x \in ?\, Q$ 的问题只需回答 $f(x) \in ?\, L$ 的问题，所以有解决 $x \in ?\, Q$ 的问题的算法如下：

(i) 用 Turing 机 T_1，在输入 $x \in A^*$ 后算出 $f(x)$. 花时间 $\leqslant p_1(|x|)$，不妨假设多项式 p_1 的系数全部为正.

(ii) 用 Turing 机 T_2，在输入 $f(x) \in A^*$ 后判断 $f(x) \in ?\, L$. 花时间 $\leqslant p_2(|f(x)|)$. 不妨假设多项式 p_2 的系数全为正.

于是问 A^* 中任一给定的 x 是否属于 A^* 中语言 Q 的问题的求解，总共只需花时间：

$$p_1(|x|) + p_2(|f(x)|) \leqslant p_1(|x|) + p_2(|x| + p_1(|x|)) \triangleq p_3(|x|)$$

(因 $|f(x)| \leqslant |x| + p_1(|x|)$).

于是问题 $x \in ?\, Q$ 是易于解算的. 由 Cook-Karp 论题 $Q \in P$. □

定理 2 设 f,g 为多项式时间可计算的部分函数,其中 g 为全函数,而 f 可为真部分函数,即可在某些点上无定义,使得

$$h(x) \triangleq f(g(x))$$

为全函数,则 h 为多项式时间可计算的全函数.

证 先观察图 3.4.

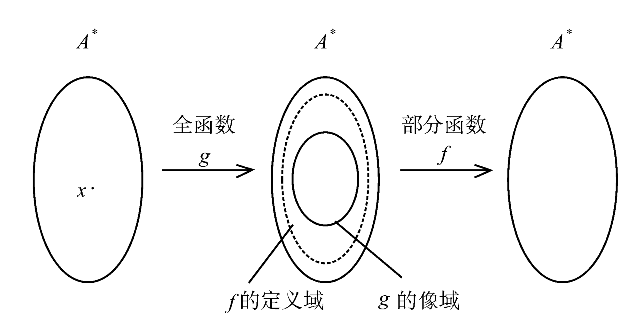


图 3.4

补充定义,说一个真部分函数 f: $A^* \to A^*$ 为多项式时间可计算的是指:

存在一台 Turing 机 T,其字母表包含 A,实现了对 f 的计算,并且伴随着一个多项式 p 使得

(i) f 的定义域中的个个字 x 都为 T 所接收,A^* 中其他的字个个都不为 T 所接收;

(ii) 对 A^* 内 f 的定义域中的每个字 x 都有

$$T \text{ 计算 } f(x) \text{ 的计算步数} \leqslant p(|x|).$$

给出补充定义后显见 $h(x)$是 Turing 可计算的全函数.因为算 g 的 Turing 机与算 f 的 Turing 机串连起来就得到了一个计算 $h(x)$的装置,由 Church–Turing论题知 $h(x)$为 Turing 可计算函数.而计算时间

$$p_1(|x|) + p_2(|g(x)|) \leqslant p_1(|x|) + p_2(|x| + p_1(|x|)) \triangleq p_3(|x|)$$

(可设 p_1, p_2 系数为正,而$|g(x)| \leqslant |x| + p_1(|x|)$).于是知 $h(x)$为多项式可计算的全函数.这里利用了 Cook–Karp 论题. □

补充定义 不确定型的 Turing 机

在第一章 § 1 中,给出了 Turing 机的定义.“我们将字母表$\{s_1,\cdots,s_n\}$及内部状态表$\{q_1,\cdots,q_m\}$以及一个满足协调条件的四重组集合联系在一起称为一部 Turing 机.”将此定义中的字“满足协调条件的”去掉即得出不确定型的 Tur-

ing 机的定义.即,我们将字母表$\{s_1,\cdots,s_n\}$及内部状态表$\{q_1,\cdots,q_m\}$以及一个四重组集合联系在一起称为一台不确定型的 Turing 机.

即在不确定型的 Turing 机中,其四重组集中可以允许有四重组对,对中的两个四重组有相同的打头二字.因此,在非确定的 Turing 机的情形,处于某一格局的机器可能找到多个合适的四重组指挥其动作,于是机器的下一个格局可能有多个不同的样子.即前一格局没有惟一地确定下一个格局.

确定型的 Turing 机是不确定型的 Turing 机的一种特殊类型.可将非确定型 Turing 机称为 Turing 机,将前者称为确定型 Turing 机.

说格局(瞬像)序列$\gamma_1,\cdots,\gamma_m$($m\geqslant 1$)是 Turing 机 M 接收字 $x\in A^*$ 的一条接收路径,是指 γ_1 是初始格局:

$$\begin{array}{c} s_0\ x \\ \uparrow \\ q_1 \end{array}$$

γ_m 是停机格局,而对任何格局 γ_i, $i\in[1,m-1]$,在 M 的四重组集中都存在一个四重组,将格局 γ_i 导向格局γ_{i+1}.即格局 γ_{i+1} 是格局 γ_i 的可能结局,序列$\gamma_1,\cdots,\gamma_m$也称为 M 对字 x 的接收计算.

定义 4　设 $A=\{s_1,\cdots,s_n\}$ 是给定的字母表, $x\in A^*$, Turing 机 M 的字母表包含 A,对 x,若存在一条 M 接收字 x 的接收路径,则称 Turing 机 M 接收字 x.如果 Turing 机 M 的字母表为 A,则

$$L=\{x\in A^*\mid M\text{ 接收 }x\}$$

称为 M 所接收的语言.

定义 5　设 $A=\{s_1,\cdots,s_n\}$, $L\subseteq A^*$,如果存在接收语言 L 的非确定型 Turing 机 M 和多项式 $p(n)$,使得对于每一个 $x\in L$,存在 M 关于 x 的接收路径$\gamma_1,\cdots,\gamma_m$,其中 $m\leqslant p(|x|)$,则称语言 L 属于 NP 类.

Cook–Karp 论题的非确定计算型式

任给字母表 A,语言 $L\subseteq A^*$,若存在着某种(不管何种)不确定计算的装置能在多项式时间内解决问题 $x\in ?\ L$,则必有 $L\in \mathrm{NP}$,即必存在一台不确定的 Turing 机 M,它能在多项式时间内解决问题 $x\in ?\ L$.

注 1　论题中的“多项式时间”,这里的时间实质上是指可能的最短的时间,即在计算中,对于 $x\in L$,机器对 x 的接收计算可能有多种路径,我们永远考虑最短(运气最好)的路径.

注 2　这个论题在 Davis 和 Weyuker 的名著中虽未明确提出,但已被隐蔽地使用.

定理 3　(i) $\mathrm{P}\subseteq \mathrm{NP}$;(ii) 若 $L\in\mathrm{NP}$,则 L 为可计算集.

证 考虑非确定型 Turing 机的定义.

$$\{q_1,\cdots,q_m\};\{s_1,\cdots,s_n\};\left\{\begin{matrix} q_i & s_j & s_k & q_l \\ & \cdots\cdots & & \\ q_\lambda & s_\mu & R & q_r \\ & \cdots\cdots & & \\ q_u & s_v & L & q_s \end{matrix}\right\}$$

在四重组集中有有穷个四重组,不限制有无两个四重组的打头二字相同,若有,则说明机器处在打头二字所示的格局时,可按甲组指示行动也可按乙组指示行动.

(i) 若 $L\in$ P 则说明存在一个确定型的 Turing 机 T,它可在多项式时间界内解决问题 $x\in ?\ L$,但此 T 也是一个非确定型的 Turing 机,于是对此 L 存在一个非确定型的 Turing 机 T,它可在多项式时间界内解决问题 $x\in ?\ L$,即 $L\in$ NP.于是,对任意 $L\in$ P 都有 $L\in$ NP,于是 P$\subseteq$NP.

(ii) 说语言 $L\in$ NP 是指存在一台非确定型的 Turing 机 M 及一个多项式 p,使得

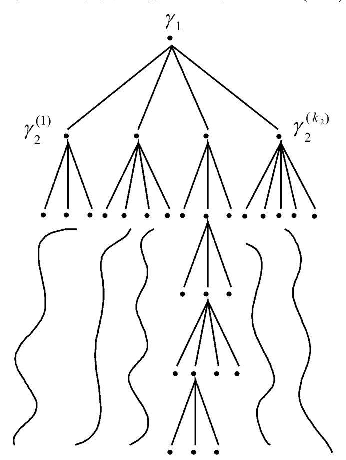
① 对任何字 $x\in L$,都存在一条 M 接收 x 的计算路径

$$\gamma_1,\gamma_2,\cdots,\gamma_N$$

初格局　　停机格局

而对任何字 $x\in A^*-L$, M 都不接收 x,即格局永远无穷延伸下去.

② 对任何字 $x\in L$,其最短接收路径长度 $\min(N)\leqslant p(|x|)$.



对任给 $x \in A^*$，问 $x \in ? L$，我们按如下步骤予以回答：

将 x 输入 M，记初格局为 γ_1，审查 M 中的四重组，挑出全部合用的四重组，将这些四重组分别作用于 γ_1，得出全部合法的二级格局 $\gamma_2^{(1)}, \gamma_2^{(2)}, \cdots, \gamma_2^{(k_2)}$.

将此树往下延伸 $P(|x|)$ 后，若在延伸的过程中至少有一个节点表示停机则知 $x \in L$；若个个节点都不表示停机，还可往下延伸，则根据定义中的②知 $x \notin L$.

这即是个判定 $x \in ? L$ 问题的完整算法，于是集 L 为可计算集. □

现在我们考虑一个特殊的语言.

令字母表 $A = \{(,),\cdot,\triangle,\vee,\wedge\}$. 令 A^* 中的字

$$(\cdot\vee\cdot\cdot\vee\ \triangle\triangle\triangle\ \vee\cdot\cdot\cdot\cdot)\wedge(\triangle\triangle\vee\triangle\vee\triangle\triangle\triangle\vee\cdot\cdot\cdot\cdot)$$

代表合取范式 $(P_1 \vee P_2 \vee \neg P_3 \vee P_4) \wedge (\neg P_2 \vee \neg P_1 \vee \neg P_3 \vee P_4)$. 合取范式 $(\neg P_2 \vee P_1) \wedge (P_3 \vee \neg P_2)$ 相应的字为 $(\triangle\triangle\vee\cdot)\wedge(\cdot\cdot\cdot\vee\triangle\triangle)$. $(\cdot\vee\triangle\triangle$ 这个字不代表合取范式；$(\cdot\vee\triangle\triangle)\wedge(\triangle\triangle\triangle\vee\cdot\cdot\triangle\triangle\triangle)$ 这个字也不代表合取范式.

于是 A^* 中的字可分为三类：不代表合取范式的字；代表不可满足的合取范式的字；代表可满足的合取范式的字. 考虑语言

$$L = \{x \in A^* \mid x \text{ 代表可满足的合取范式}\},$$

我们可以问语言 $L \in ? \mathrm{P}$，语言 $L \in ? \mathrm{NP}$.

先看这样表示的合取范式的长度问题. 例如：

$$x = (P_1 \vee P_2 \vee \neg P_3 \vee P_4) \wedge (\neg P_2 \vee \neg P_1 \vee \neg P_3 \vee P_4) \wedge (\neg P_2 \vee \neg P_1 \vee \neg P_3),\ |x| = 27;$$

$$x' = (\cdot\vee\cdot\cdot\vee\triangle\triangle\triangle\ \vee\cdot\cdot\cdot\cdot)\wedge(\triangle\triangle\vee\triangle\vee\triangle\triangle\triangle\vee\cdot\cdot\cdot\cdot)\wedge(\triangle\triangle\vee\triangle\vee\triangle\triangle\triangle),$$

$|x'| = 27 + 15 = 42$.

$|x| \leqslant |x'|$，所以若由 x 方式表示的可满足合取范式的全体 $\in P$，则

$$\text{由 } x' \text{ 方式表示的可满足合取范式的全体} \in \mathrm{P}.$$

现在问，若由 x' 方式表示的可满足合取范式的全体 $\in \mathrm{P}$，则

$$\text{由 } x \text{ 方式表示的可满足合取范式的全体是否一定仍} \in \mathrm{P}?$$

考虑到 $|x| \geqslant \nu$（此中 $\nu = x$ 中出现的不同命题变元的个数），知 $|x| \leqslant |x'| \leqslant \nu \cdot |x| \leqslant |x|^2$. 于是若计算时间 $\leqslant p(|x'|)$，必有计算时间 $\leqslant p(|x|^2) = p_1(|x|)$. 这里利用了 $|x|^2 \geqslant |x'|$，可以假设多项式 p 的系数全为正.

于是可知，合取范式的可满足性问题是否 $\in \mathrm{P}$，不依问题的表示方案而变.

即这个问题具有客观的意义.这一点在证明与 NP 理论有关的事情时是有重要意义的.我们记 A^* 中所有代表可满足的合取范式的字所构成的集合为 SAT.现今的科学水平还不知道是否 SAT $\in$ P.

以下为证明 SAT $\in$ NP 作点准备.

考虑字母表 $C_n=\{\alpha_1,\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n,/\}$,其中 n 为任一给定的正整数.于是字母表 C_n 上的任一以/打头的行都代表一合取范式,如 $/\alpha_1\alpha_2\bar{\alpha}_3\alpha_4/\bar{\alpha}_2\bar{\alpha}_1\bar{\alpha}_3\alpha_4/\bar{\alpha}_2\bar{\alpha}_1\bar{\alpha}_3$ 代表

$$(P_1\vee P_2\vee\neg P_3\vee P_4)\wedge(\neg P_2\vee\neg P_1\vee\neg P_3\vee P_4)\wedge(\neg P_2\vee\neg P_1\vee\neg P_3);$$

$/\alpha_1\bar{\alpha}_2$ 代表$(P_1\vee\neg P_2)$;/代表一个空子句;

///代表空子句 $\wedge$ 空子句 $\wedge$ 空子句;

$/\alpha_1\bar{\alpha}_2/\alpha_3$ 代表$(P_1\vee\neg P_2)\wedge P_3$;

$/\alpha_1\alpha_2/$代表$(P_1\vee P_2)\wedge$空子句;

$/\bar{\alpha}_1\alpha_2//$代表$(\neg P_1\vee P_2)\wedge$空子句$\wedge$空子句;

$/\bar{\alpha}_1\alpha_2///\alpha_2\bar{\alpha}_3$ 代表$(\neg P_1\vee P_2)\wedge$空子句$\wedge$空子句$\wedge(P_2\vee\neg P_3)$.

现在令 $\mathrm{SAT}_n=\{x\in C_n^*\mid x$ 代表可满足的合取范式$\}$,SAT_n 的意味是一切命题变元个数$\leqslant n$ 的可满足的合取范式所构成的集合.

定理 4 对于任意给定的正整数 n,$\mathrm{SAT}_n\in$ P.

证 构造出 Turing 机 M,使其字母表包含 C_n,并且对 C_n^* 中的任一字 x,若 $x\in\mathrm{SAT}_n$,则 M 在多项式时间内接收 x,若 $x\notin\mathrm{SAT}_n$,则 M 不接收 x.

M 的字母表为:$\tilde{C}_n=\{\alpha_1,\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n,/,|\}$;

M 的状态表为:$\{q_1,m_j,r_i^+,r_i^-,v_i^+,v_i^-,w_i^+,w_i^-,p,\bar{p}\}$,

其中 $j=1,2,\cdots,n+1$;$i=1,2,\cdots,n$.即状态表中共有 $7n+4$ 个内部状态.检查输入的 x 是否以/号打头用下列诸四重组(行为规则):

$q_1\quad B\quad R\quad q_1$

$q_1\quad /\quad L\quad m_1$

$q_1\quad S\quad L\quad q_1$ 其中 S 遍取$\alpha_1,\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n$.

$\cdots\ s_0\ \underset{\substack{\uparrow\\ q_1}}{s_0}\ /\alpha_2\bar{\alpha}_3\cdots\qquad\cdots\ s_0\ \underset{\substack{\uparrow\\ q_1}}{s_0}\ \alpha_3\bar{\alpha}_1\cdots$　　q_1BRq_1 为见空右移;

$\cdots\ s_0\ s_0\ \underset{\substack{\uparrow\\ q_1}}{/}\ \alpha_2\bar{\alpha}_3\cdots\qquad\cdots\ s_0\ s_0\ \underset{\substack{\uparrow\\ q_1}}{\alpha_3}\ \bar{\alpha}_1\cdots$　　q_1SLq_1 为见到的非为/就左移,然后循环往复至无穷;

$\cdots\ s_0\ \underset{\substack{\uparrow\\ m_1}}{s_0}\ /\alpha_2\bar{\alpha}_3\cdots\qquad\cdots\ s_0\ \underset{\substack{\uparrow\\ q_1}}{s_0}\ \alpha_3\bar{\alpha}_1\cdots$

$q_1 \; / \; L \; m_1$ 为见到 / 就左移回去，开始判定工作，处于“开始判定”的内态.

检查带上有无“/”号剩下用下列诸四重组：

$m_{n+1}\quad B\quad R\quad p$

$p\quad S\quad R\quad p$　其中 S 遍取 $\alpha_1,\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n,|$.

$p\quad /\quad L\quad \bar{p}$

$\bar{p}\quad S\quad R\quad p$　其中 S 遍取 $\alpha_1,\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n,/,|,s_0$.

$\cdots\ s_0\ \ s_0\ \times\ \times\ \times\ \times\ /\times\ \times\ \times\ \times\ s_0\ s_0\cdots$
　　↑
　m_{n+1}

$\cdots\ s_0\ s_0\ \times\ \times\ \times\ \times\ /\times\ \times\ \times\ \times\ s_0\ s_0\cdots$
　　　↑
　　　p

$\cdots\ s_0\ s_0\times\ \times\ \times\ \times\ \ /\ \times\ \times\ \times\ \times\ s_0\ s_0\cdots$
　　　　　　　↑
　　　　　　　p

$\cdots\ s_0\ s_0\times\ \times\ \times\ \ \times\ /\times\ \times\ \times\ \times\ s_0\ s_0\cdots$
　　　　　　↑
　　　　　　$\bar{p}$

$\cdots\ s_0\ s_0\times\ \times\ \times\ \times\ \ /\ \times\ \times\ \times\ \times\ s_0\ s_0\cdots$
　　　　　　　↑
　　　　　　　p

回到了上面的格局，无穷循环下去.

$\cdots\ s_0\ \ \ s_0\ \times\ \times\ \times\ \times\ \times\ \times\ s_0\ s_0\cdots$
　　↑
　m_{n+1}

$\cdots\ s_0\ s_0\ \times\ \times\ \times\ \times\ \times\ \times\ s_0\ s_0\cdots$
　　　↑
　　　p

$\cdots\ s_0\ s_0\times\ \times\ \times\ \times\ \times\ \times\ \ s_0\ s_0\cdots$
　　　　　　　　↑
　　　　　　　　p

停机，四重组中无 ps_0 打头.

算法的思路：

0. 若首字符不是 /，则走去死循环；若是 /，则开始判定计算.

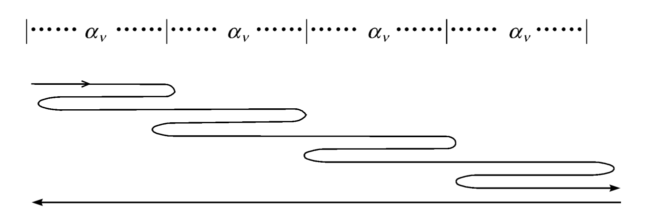
1. 对第 1 个命题变元随机选真假值，若选出为真，则将各子句中凡出现 α_1 的句子之首的 / 号改为 | 号；若选出为假，则将各子句中凡出现 $\bar{\alpha}_1$ 的句子之首的 / 号改为 | 号，此意义是此子句已得到了满足，标以记号. 回到字首.

2. 对第 2、…、第 n 个命题变元重复步 1.

3. 观察被改造过了的字 x，（改动仅可能为将 / 改为 |），若无 / 号了，即所有 / 号全被改为了 | 号，表示在所处随机赋定了值的指派之下，个个子句为真，即 x

可满足,于是该机器停止;若还有 / 存在则让机器去死循环.

4. 复杂度之估计.



以上为走针之最费时之情形(赋值之选取已使合取范式为真之条件下).

由于走头需 2 步,走尾需 $|x|$ 步,共 $|x|+2$ 步,总共花时间为

$$\text{时界} \leqslant 4n|x|+|x|+2.$$

n 取常数时此界为线性,一次多项式.当视 n 为变数时,

$$\text{时界} \leqslant 4n|x|+|x|+2 \leqslant 4|x|^2+|x|+2 \ (\text{因}\ n \leqslant |x|),$$

时界亦为多项式,于是知 $\mathrm{SAT}_n \in \mathrm{NP}$. □

下面列出 Turing 机 M 的全部四重组.

$q_1 \quad B \quad R \quad q_1$
$q_1 \quad / \quad L \quad m_1$
$q_1 \quad S \quad L \quad q_1$ S 遍取 $\alpha_1,\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n$,
} 作查头之用;

$m_i \quad B \quad R \quad r_i^+$
$m_i \quad B \quad R \quad r_i^-$
} 为 α_i 选真假值, $i=1,2,\cdots,n$;

$r_i^\pm \quad | \quad R \quad v_i^\pm$ 当前子句已满足,观察下一子句;

$r_i^\pm \quad / \quad R \quad r_i^\pm$ 当前子句尚未发现满足,寻求满意文字;

$r_i^+ \quad S \quad R \quad r_i^+$ S 遍取 $\alpha_1,\cdots,\alpha_{i-1},\alpha_{i+1},\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n$,看到的不满意就右移;

$r_i^- \quad S \quad R \quad r_i^-$ S 遍取 $\alpha_1,\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_{i-1},\bar{\alpha}_{i+1},\cdots,\bar{\alpha}_n$,看到的不满意就右移;

$r_i^+ \quad \alpha_i \quad L \quad w_i^+$
$r_i^- \quad \bar{\alpha}_i \quad L \quad w_i^-$
} 发现了满意文字后,装新内态 $w_i^\pm$;

$w_i^\pm \quad S \quad L \quad w_i^\pm$ S 遍取 $\alpha_1,\cdots\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n$,
$w_i^\pm \quad / \quad | \quad v_i^\pm$
} 标志被满足了的子句;

$v_i^\pm \quad S \quad R \quad v_i^\pm$ S 遍取 $\alpha_1,\cdots,\alpha_n,\bar{\alpha}_1,\cdots,\bar{\alpha}_n,|$,
$v_i^\pm \quad / \quad R \quad r_i^\pm$
} 寻找下一个未被满

足之子句;

$r_i^{\pm}$ B L m_{i+1}
$v_i^{\pm}$ B L m_{i+1} } 准备为下一个命题变元选真假值;

m_{i+1} S L m_{i+1} S 遍取$\alpha_1, \cdots, \alpha_n, \bar{\alpha}_1, \cdots, \bar{\alpha}_n$, /, |, 找输入字 x 的头.

各态($i = 1, 2, \cdots, n$)的意义为

m_i 为寻求字头态,即左行到字头态,也是选命题变元真假值态.

$r_i^{\pm}$ 为寻求满意文字态.

$v_i^{\pm}$ 为寻求未被满足之子句态.

$w_i^{\pm}$ 为去标志本子句已被满足之态.

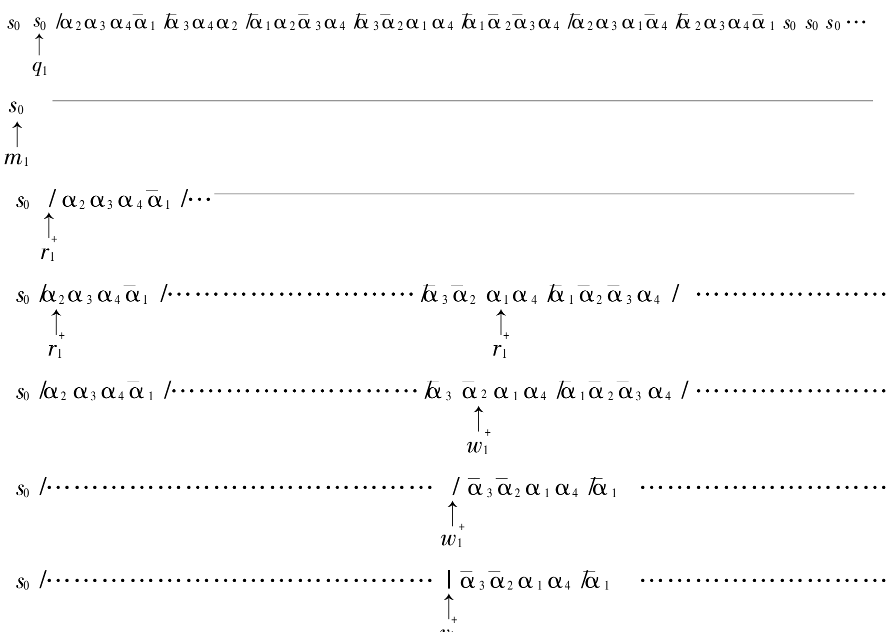
m_{n+1} 为判断本合取范式是否已被满足之态.

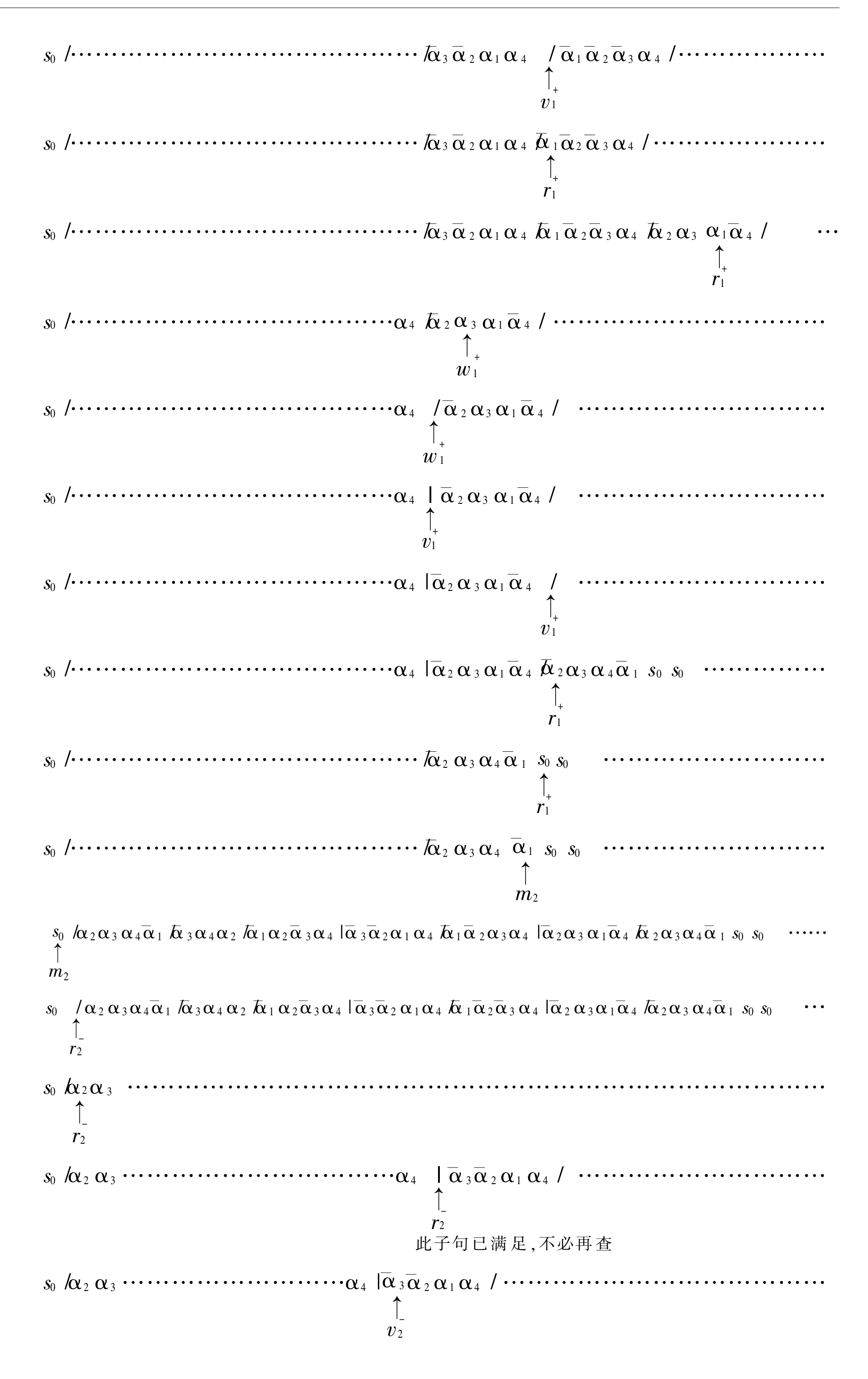
m_{n+1} B R p
p S R p S 遍取$\alpha_1, \cdots, \alpha_n, \bar{\alpha}_1, \cdots, \bar{\alpha}_n$, |,
p / L $\bar{p}$
$\bar{p}$ S R p S 遍取$\alpha_1, \cdots, \alpha_n, \bar{\alpha}_1, \cdots, \bar{\alpha}_n$, /, |, B, } 最后总查是否还残存"/"符,即是否满足了.

若无"/"符,即满足了,则停机;

若有"/"符,即未满足,则发散↑.

Turing 机 M 走机之一例




此子句已满足,不必再查

s_0 /$\alpha_2\alpha_3$ ……………………… α_4 |$\bar{\alpha}_3\bar{\alpha}_2\alpha_1\alpha_4$ /$\bar{\alpha}_1\bar{\alpha}_2\bar{\alpha}_3\alpha_4$ |$\bar{\alpha}_2$ ………………
↑_
v_2

s_0 /$\alpha_2\alpha_3$ …………………………………… /$\bar{\alpha}_1\bar{\alpha}_2\bar{\alpha}_3\alpha_4$ |$\bar{\alpha}_2$ ………………
↑_
r_2

s_0 /$\alpha_2\alpha_3$ …………………………………… /$\bar{\alpha}_1$ $\bar{\alpha}_2\bar{\alpha}_3\alpha_4$ |$\bar{\alpha}_2$ ………………
↑_
r_2

s_0 /$\alpha_2\alpha_3$ …………………………………… /$\bar{\alpha}_1$ $\bar{\alpha}_2\bar{\alpha}_3\alpha_4$ |$\bar{\alpha}_2$ ………………
↑_
w_2

s_0 /$\alpha_2\alpha_3$ …………………………………… / $\bar{\alpha}_1\bar{\alpha}_2\bar{\alpha}_3\alpha_4$ |$\bar{\alpha}_2$ ………………
↑_
w_2

s_0 /$\alpha_2\alpha_3$ …………………………………… | $\bar{\alpha}_1\bar{\alpha}_2\bar{\alpha}_3\alpha_4$ |$\bar{\alpha}_2$ ………………
↑_
v_2

s_0 /$\alpha_2\alpha_3$ ……………………… |$\bar{\alpha}_1\bar{\alpha}_2\bar{\alpha}_3\alpha_4$ |$\bar{\alpha}_2\alpha_3\alpha_1\bar{\alpha}_4$ /$\bar{\alpha}_2\alpha_3\alpha_4\bar{\alpha}_1$ s_0 s_0 ………
↑_
v_2

s_0 /$\alpha_2\alpha_3$ …………………………………… /$\bar{\alpha}_2\alpha_3\alpha_4\bar{\alpha}_1$ s_0 s_0 ………………
↑_
r_2

s_0 /$\alpha_2\alpha_3$ …………………………………… / $\bar{\alpha}_2\alpha_3\alpha_4\bar{\alpha}_1$ s_0 s_0 ………………
↑_
w_2

s_0 /$\alpha_2\alpha_3$ …………………………………… | $\bar{\alpha}_2\alpha_3\alpha_4\bar{\alpha}_1$ s_0 s_0 ………………
↑_
v_2

s_0 /$\alpha_2\alpha_3$ …………………………………… |$\bar{\alpha}_2\alpha_3\alpha_4\bar{\alpha}_1$ s_0 s_0 ………………
↑_
v_2

s_0 /$\alpha_2\alpha_3$ …………………………………………… |$\bar{\alpha}_2\alpha_3\alpha_4$ $\bar{\alpha}_1$ s_0 s_0 ………
↑
m_3

s_0 /$\alpha_2\alpha_3\alpha_4\bar{\alpha}_1$ /…………………………………… |$\bar{\alpha}_2\alpha_3\alpha_4\bar{\alpha}_1$ s_0 s_0 ………………
↑
m_3

定义 6　设 L, Q 为 A^* 上的语言,如果存在从 A^* 到 A^* 的多项式可计算的全函数 f,使得 $x \in Q \Leftrightarrow f(x) \in L$,则我们称 Q 多项式时间可归约到 L,并记作 $Q \underset{p}{\leqslant} L$.

定理 5　若 $R \underset{p}{\leqslant} Q$ & $Q \underset{p}{\leqslant} L$,则 $R \underset{p}{\leqslant} L$.

证 见图 3.5,考虑将 $x \in ?\ R$ 的问题归结为 $g(f(x)) \in ?\ L$ 的问题在实际操作中所花的时间.

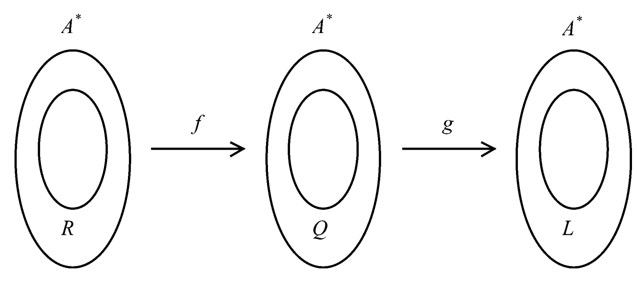


图 3.5

(i) 复合函数 $g(f(x))$ 为全函数,为可计算的,为多项式时间可计算的;

(ii) $x \in R \Leftrightarrow f(x) \in Q \Leftrightarrow g(f(x)) \in L$. □

定义 7 语言 L 称作 NP 难度的,是指对任一语言 $Q \in \mathrm{NP}$ 都有 $Q \underset{p}{\leqslant} L$;如果 L 为 NP 难度的且 $L \in \mathrm{NP}$,则称 L 是 NP 完全的.

注 1 说 L 为 NP 难度的是指,若 $x \in ?\ L$ 的问题能很快地解决则对任何属于 NP 的 Q 而言,$x \in ?\ Q$ 的问题都能很快地解决.

即若有算法克服 L,则有算法克服 NP 中的任一集 Q;

即问题 L 的难度大于等于 NP 中的任一问题 Q 难度.

注 2 说 L 为 NP 完全的是指 L 是 NP 中的问题中的最难者(图 3.6).

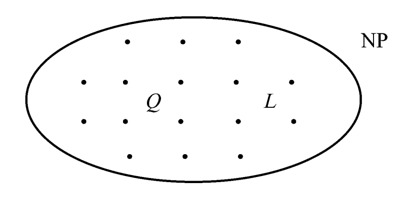


图 3.6

定理 6 如果有一个 NP 完全的语言 L,它属于 P,则 NP = P.

证 设 $Q \in \mathrm{NP}$,往下证 $Q \in \mathrm{P}$,由于 L 为 NP 难度的,于是有 $Q \underset{p}{\leqslant} L$,于是有 A^* 到 A^* 上的多项式时间可计算的全函数 f,使得

$$x \in Q \Leftrightarrow f(x) \in L.$$

于是存在一个确定的算法,它可在多项式时间内判定 A^* 中任一给定的 x 是否属于 Q 的问题,于是据 Cook–Karp 论题知 $Q \in \mathrm{P}$. □

注 定理 6 说明对任一给定的 NP 完全问题,若它 $\in$ P 则 P = NP,若它不属

于P则 $P \neq NP$.于是解决 $P = ?\ NP$ 的问题可以从任一给定的NP完全问题着手.

§3. Cook定理

定理1(Cook) SAT为NP完全的语言.

证 在本章§2的定理4的证明中,我们已证 $SAT_n \in NP$,其中 n 为任一给定的正整数,将这证明加以改进可证明出 $SAT \in NP$.不过,利用关于不确定计算的Cook–Karp论题可以看出 $SAT \in NP$.看法如下:

任给一合取范式 $x \in SAT$,求出 x 中所含命题变元,设为 $p_1, \cdots, p_n$.随机地为这 n 个命题变元指派好真假值.对指派好了真假值的 x 计算其真假值.若算出为真则停机,输出讯号:"可满足";若算出为假则作死循环.

对以上算法显然有:$x \in SAT \Leftrightarrow$ 算法会在某种情况下停止;若 $x \in SAT$ 则算法时间被一次多项式所囿界,如是,按Cook–Karp论题知 $SAT \in NP$.

现在只需证明SAT为NP难度的,即要证明,若 $L \in NP$ 则 $L \underset{p}{\leqslant} SAT$.

说语言 $L \in NP$ 是指存在一台非确定型的Turing机 M,及一个多项式 p,使得

① L 是某字母表 A 上的某些有穷串的集,即 $L \subseteq A^*$;

A = Turing机 M 的字母表$\{s_1, \cdots, s_r\}$;机器 M 的态表为$\{q_1, \cdots, q_m\}$.

② 对任何字 $u \in L$,都存在一条 M 接收 u 的路径 $\gamma_1, \cdots, \gamma_n$;而对任何字 $u \in A^* - L$, M 不接收 u.

③ 对任何字 $u \in L$,其最短接收路径长 $\min(n) \leqslant p(|u|)$.

我们证明存在一个多项式时间可计算函数,它将 A^* 中的任一个行(字) u 转换成一个合取范式 δ_u,使得 u 被 M 接收 $\Leftrightarrow \delta_u$ 为可满足.

对任一 u,记 $p(|u|) = t$,可不妨假设 $p(|u|)$ 恒 $\geqslant |u|$.我们知道,对 A^* 中任一 u,若 $u \in L$,则 M 有一条在 t 步内接收 u 的路,若 $u \notin L$,则 M 在输入 u 后走任何路都不会接收 u,即不停机,因此为考虑 $u \in ?\ L$ 的问题,我们只需考虑 M 在输入 u 后的步数 $\leqslant t$ 的一切可能的计算路,看这些路中是否有一条已有停机端点格局.

又因为 M 每算一步左右移至多只一格,于是对这步数 $\leqslant t$ 的一切可能路, M 机的指针左右移至多只有 t 格.由于 $t \geqslant |u|$,我们只需考虑 M 机的带子上的 $2t+1$ 个格子(图3.7).

由于只需考虑 t 步动作,我们只需 $t+1$ 条纸带(每条纸带上 $2t+1$ 个格子),即可展示所需观察的全部计算过程.即只需花一张$(t+1)\times(2t+1)$个格子

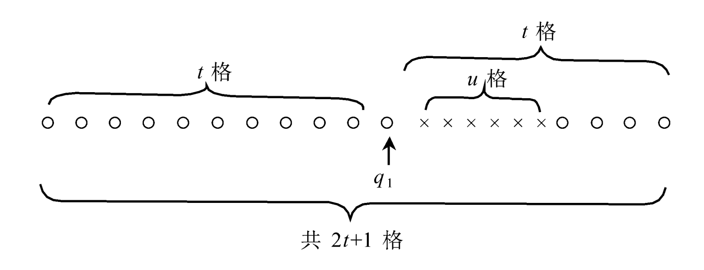


图 3.7

的长方格纸即可展现全部计算(图 3.8).

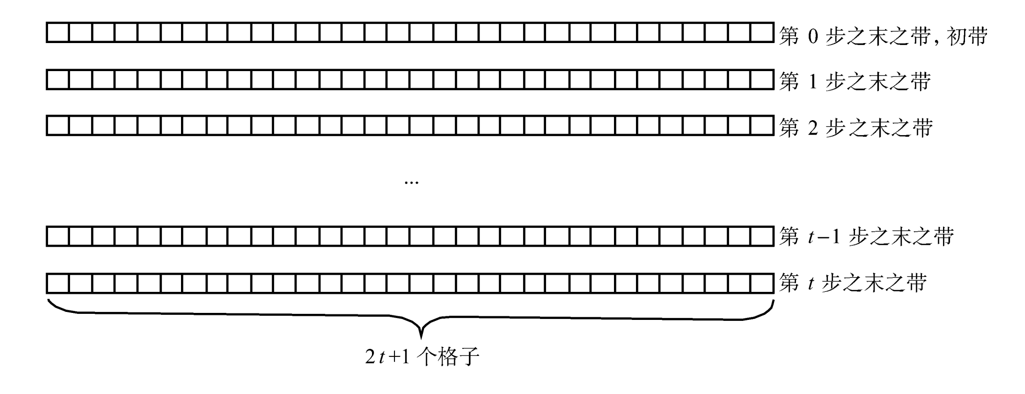


图 3.8

初带的形状为 $s_0^{t+1}us_0^{t-|u|}$, M 机处于 q_1 态,指针指着第 $t+1$ 个符号.

若 $u\in L$,则输入后,可能在第 t 步之末以前已达到了接收态 q_m.若在我们接收 L 的机器 M 中加上"踏步不前"的四重组 $q_i s_j s_j q_i$,其中 $i=m$, $j=1,2,\cdots,r$,则事态发生如下变化:若某 $u\in L$ 输入后在第 t 步之末以前达到接收态 q_m,将此终止格局直延至第 t 步之末,则 u 在输入后第 t 步之末达到接收态 q_m.于是对此新的 Turing 机,只要 $u\in L$, u 必有在第 t 步之末达接收态 q_m 之计算路;若 $u\notin L$,则 u 必无在第 t 步之末达接收态 q_m 之计算路.因此

$$u\in L\Leftrightarrow u \text{ 有在 } t \text{ 步}(p(|u|)\text{ 步})\text{ 之末达接收态 } q_m \text{ 之计算路}.$$

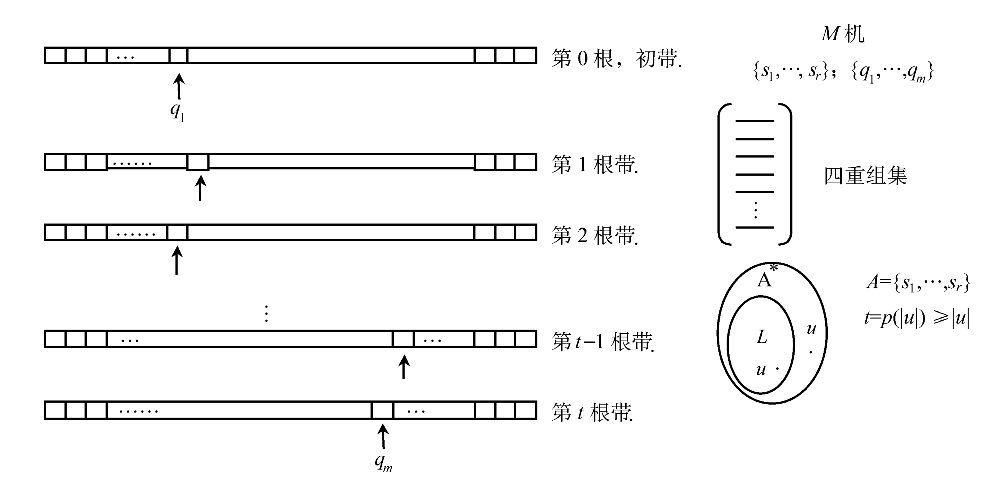
于是可以在一个统一的时间上来观察问题,给我们想问题带来了方便.

而句子"u 有在第 $p(|u|)$ 步之末达接收态 q_m 之计算路",在写透之后就是一个合取范式 δ_u,其长度为 $|u|$ 的多项式.并且,话真$\Leftrightarrow\delta_u$ 可满足,于是知

$[u \in L \Leftrightarrow \delta_u$ 可满足$] \wedge [$由 u 得出 δ_u 只花 $|u|$ 的多项式时间$]$.

于是 $L \leqslant_p \mathrm{SAT}$.这里 L 为任一给定的属于 NP 的语言,于是 SAT 为 NP 难度的.前面已知 SAT $\in$ NP,于是知 SAT 为 NP 完全的.

以下将上述路线细化,完成出来.



命题变元集

$\rho_{h,j,k}$：第 k 根带上指针指着第 j 个位置且内态为 q_h

$\delta_{i,j,k}$：第 k 根带上的第 j 个格子上符号为 s_i

$h = 1,\cdots,m$；$i = 0,1,\cdots,r$；

$k = 0,1,\cdots,t$；

$j = 1,2,\cdots,2t+1$.

共 $m \times (2t+1) \times (t+1)$ 个变元　　　　共 $(r+1) \times (2t+1) \times (t+1)$ 个变元

利用这些命题变元作为语言原子,我们可将语句“$u \in L$”即“u 输进 M 机后在第 t 步($p(|u|)$步)之末达到接收态 q_m 之计算路”完全写清楚如下:

为作准备,将话“变量值 $x_1, x_2, \cdots, x_l$ 中恰有一个为真”写成如下合取范式:

$$(x_1 \vee x_2 \vee \cdots \vee x_l) \wedge (\bar{x}_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_3) \wedge \cdots$$

$$\wedge (\bar{x}_1 \vee \bar{x}_l) \wedge \cdots \wedge (\bar{x}_{l-1} \vee \bar{x}_l) \triangleq \Delta\{x_1, x_2, \cdots, x_l\}$$

此 CNF 之长度为 $O(l^2)$.

“$u \in L$”即有第 0,第 1,⋯,第 t 根幻灯片存在,使第 t 根为接收态,第 0 根

为初态,且第 1 至第 t 根带,每带都是前一带之按 M 指挥之合法发展.

有第 0,第 1,…,第 t 根幻灯片存在,即:①每根片上指针恰指着一个位置,恰有一个内态 q;②每根片的每一格上恰有一个符号 s.

每根片上指针恰指一个位置,恰有一个内态,即

$$\bigwedge_{0\leq k\leq t}\Delta\left\{\rho_{h,j,k}\;\middle|\;\begin{array}{l}h=1,\cdots,m\\ j=0,1,\cdots,2t+1\end{array}\right\},\tag{3.1}$$

此式之长度为 $O(t^3)$.

每根片的每一格上恰有一个符号 s,即

$$\bigwedge_{0\leq k\leq t}\ \bigwedge_{1\leq j\leq 2t+1}\Delta\{\delta_{i,j,k}\mid i=0,1,\cdots,r\},\tag{3.2}$$

此式之长度为 $O(t^2)$.

第 0 根幻灯片为初格局,如图 3.9 所示.

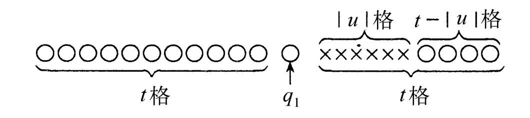


图 3.9

$$\bigwedge_{0\leq j\leq t+1}\delta_{0,j,0}\ \bigwedge_{1\leq j\leq |u|}\delta_{u_j,t+1+j,0}\ \bigwedge_{1\leq j\leq t-|u|}\delta_{0,t+1+|u|+j,0}\wedge\rho_{1,t+1,0}\tag{3.3}$$

式中 u_j 为字 u 的第 j 个符号的标号,即 $u=s_{u_1}s_{u_2}\cdots s_{u_{|u|}}$.此式之长度为 $O(t)$.

第 t 根幻灯片为接收态,如图 3.10 所示.

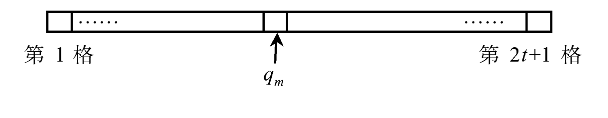


图 3.10

$$\bigvee_{1\leq j\leq 2t+1}\rho_{m,j,t}\tag{3.4}$$

此式之长度为 $O(t)$.

最后,第 1 根,第 2 根,…,第 t 根带每根都是前一带按机器 M 指挥之合法发展.此话是要说清带上符号如何变化,指针位置如何变化及内部状态 q 如何变化,即格局如何随时间变化.

带上符号之变化法则分二种情形:

(i) 现在未被指针指着的格子,其上之符号不变,即下一时刻此格上之符号相同于目前此格上之符号;

(ii) 现在正被指针指着的格子,其上符号之变化方式听机器 M 之某四重组指挥.

针位置及内态 q 之变化方式听机器 M 之某四重组指挥.

设机器 M 的四重组集为

$$\{q_{i_a}s_{j_a}s_{k_a}q_{l_a} \mid a = 1,\cdots,\bar{a}\}, \tag{a}$$

$$\{q_{i'_b}s_{j'_b}Rq_{l'_b} \mid b = 1,\cdots,\bar{b}\}, \tag{b}$$

$$\{q_{i''_c}s_{j''_c}Lq_{l''_c} \mid c = 1,\cdots,\bar{c}\}, \tag{c}$$

$$1 \leqslant i_a, i'_b, i''_c, l_a, l'_b, l'_c \leqslant m,$$

$$0 \leqslant j_a, j'_b, j''_c, k_a \leqslant r.$$

于是从第 k 根带到第 $k+1$ 根带之格局之演化由下式表达:

$$\bigwedge_{0\leqslant k\leqslant t-1, 1\leqslant j\leqslant 2t+1}\left\{\begin{array}{ll}\bigvee_{0\leqslant v\leqslant r}\left[(\delta_{v,j,k}\wedge\delta_{v,j,k+1})\bigwedge_{1\leqslant h\leqslant m}(\neg\rho_{h,j,k})\right] & \text{指针不指在第 } j \text{ 格上}\\ \bigvee_{1\leqslant a\leqslant\bar{a}}\delta_{j_a,j,k}\wedge\rho_{i_a,j,k}\wedge\delta_{k_a,j,k+1}\wedge\rho_{l_a,j,k+1} & \text{指针指着第 } j \text{ 格,(a) 起作用}\\ \bigvee_{1\leqslant b\leqslant\bar{b}}\delta_{j'_b,j,k}\wedge\rho_{i'_b,j,k}\wedge\delta_{j'_b,j,k+1}\wedge\rho_{l'_b,j+1,k+1} & \text{指针指着第 } j \text{ 格,(b) 起作用}\\ \bigvee_{1\leqslant c\leqslant\bar{c}}\delta_{j''_c,j,k}\wedge\rho_{i''_c,j,k}\wedge\delta_{j''_c,j,k+1}\wedge\rho_{l''_c,j-1,k+1} & \text{指针指着第 } j \text{ 格,(c) 起作用}\end{array}\right\}, \tag{3.5}$$

此花括号内之式为析取范式,可用分配律展开成合取范式,所得合取范式之长度为常数,因此整个(3.5)式为合取范式,且长度为 $O(t^2)$.

最后得

$$\delta_u \triangleq (3.1)\wedge(3.2)\wedge(3.3)\wedge(3.4)\wedge(3.5),$$

式子长度为 $O(t^3)$,且 $u\in L\Leftrightarrow\delta_u\in \mathrm{SAT}$.

于是 $L\leqslant_p \mathrm{SAT}$,即 SAT 为 NP 难度的. □

§4. 另外几个 NP 完全问题

定理 1 设 Q 是 NP 完全问题, $Q\leqslant_p L$,则 L 是 NP 难度的.

证 设 $R \in \mathrm{NP}$.因 Q 为 NP 完全,于是有 $R \underset{p}{\leqslant} Q \underset{p}{\leqslant} L$,于是 $R \underset{p}{\leqslant} L$.即对任何 $R \in \mathrm{NP}$ 有 $R \underset{p}{\leqslant} L$,即 L 是 NP 难度的. □

推论 1 设 Q 为 NP 完全问题, $L \in \mathrm{NP}$ 且 $Q \underset{p}{\leqslant} L$ 则 L,为 NP 完全的.

定理 2 3–SAT 是 NP 完全问题.

证 显然,3–SAT $\in$ NP,只需证明 3–SAT 是 NP 难的.

引理 $(A \vee B) \wedge Q \xlongequal{\text{等可满足性于}} (A \vee p) \wedge (\bar{p} \vee B) \wedge Q$,

其中 A, B, Q 为命题逻辑中的表达式,它们中不出现命题变元 p; p 为独立的命题变元.

证 ①在某赋值之下,右式为真则左式必为真;

②在某赋值之下,左式为真,例如 A 真,则命 p 为假后得右式为真. □

据引理有,例如 $\xi = (p_1 \vee \bar{p}_2 \vee p_3 \vee \bar{p}_4 \vee p_5 \vee \bar{p}_6) \wedge Q(p_1, p_2, \cdots, p_{100})$为合取范式,则有

$$\begin{aligned}
\xi &= (p_1 \vee \bar{p}_2 \vee u_1) \wedge (\bar{u}_1 \vee p_3 \vee \bar{p}_4 \vee p_5 \vee \bar{p}_6) \wedge Q(p_1, p_2, \cdots, p_{100}) \\
&= (p_1 \vee \bar{p}_2 \vee u_1) \wedge (\bar{u}_1 \vee p_3 \vee u_2) \wedge (\bar{u}_2 \vee \bar{p}_4 \vee p_5 \vee \bar{p}_6) \\
&\quad \wedge Q(p_1, p_2, \cdots, p_{100}) \\
&= (p_1 \vee \bar{p}_2 \vee u_1) \wedge (\bar{u}_1 \vee p_3 \vee u_2) \wedge (\bar{u}_2 \vee \bar{p}_4 \vee u_3) \\
&\quad \wedge (\bar{u}_3 \vee p_5 \vee \bar{p}_6) \wedge Q(p_1, p_2, \cdots, p_{100}) \\
&= \cdots\cdots = \xi',
\end{aligned}$$

于是,$\xi \xlongequal{\text{等可满足性于}} \xi'$,其中$|\xi'| \leqslant 3|\xi|$,$\xi'$为子句长$\leqslant 3$ 的合取范式.

于是,若 3–SAT 有多项式型的判别算法则 SAT 就有了多项式型的判别算法.

即 SAT $\underset{p}{\leqslant}$ 3–SAT.即 3–SAT 为 NP 难的. □

所谓完全子图问题是,任给定一个图及正整数 K,问此图中有 K 阶完全子图吗?即问此图中是否有这样的 K 个点,其中每两个点皆存在着边相连吗?(图 3.11)

定理 3 完全子图问题是 NP 完全的.

例 G: $V = \{1,2,3,4,5,6\}$,

$E = \{12,13,14,15,24,34,56\}$.

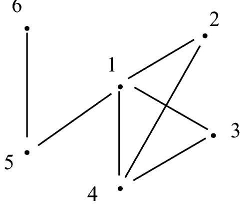


图 3.11

定理 3 的证 一、完全子图问题 $\in$ NP.

设 $V = \{1,2,\cdots,n\}$; $E = \{\cdots\cdots\}$,问有 K 阶完全子图吗?

一个十分自然的不确定型算法如图 3.12 所示.

点 1	点 2	点 3	点 4	……	点 n
要	要	要	要		要
否	否	否	否		否
?	?	?	?		?

针对随机地要下来了的点进行检验,看这些

点是否构成了 K阶完全子图的顶点

图 3.12

计算时间:$O(n)+O(K^2)\sim O(n)+O(n^2)\sim O(n^2)$小于等于问题长度的二次多项式. □

二、完全子图问题是 NP 难度的.

我们证明合取范式的可满足性问题可多项式地归结为完全子图问题.即有一个将合取范式 CNF 转换成图 G 及自然数 K 的转换,使得

$$\left[\ CNF\gamma \xrightarrow[\text{转换得出}]{\text{用多项式时间}} \text{图 } G \text{ 和数 } K, \gamma \text{ 可满足} \Leftrightarrow G \text{ 有 } K \text{ 阶完全子图}\right].$$

设 $\gamma=\gamma_1\wedge\gamma_2\wedge\cdots\wedge\gamma_k$(图 3.13).令:$K=k$,即 K 等于合取范式中所含有的析取子句的个数;$G=(V,E)$,其中

$V=\{(\alpha,i)\mid\alpha$ 为γ_i 中之一文字$\}$,即 γ 中出现过多少次文字就有多少个顶点.

$E=\{((\alpha,i),(\beta,j))\mid\alpha\neq\beta$ 且 $i\neq j\}$,即两个出现在不同子句的文字,若不是互反则用边相连.

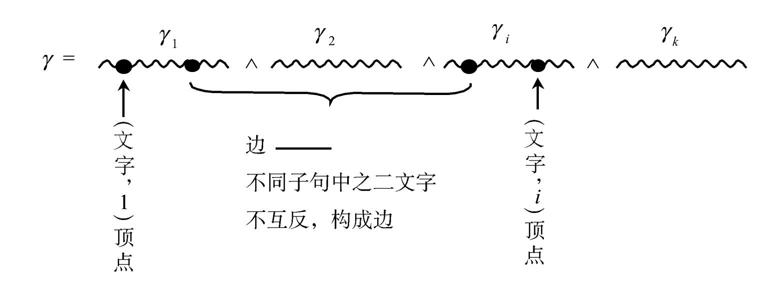


图 3.13

对如此构成的图有:

(i) γ 可满足则图 G 有 K 阶完全子图;

(ii) 若图 G 有 K 阶完全子图,则 γ 可满足. □

我们只需观察清楚如下简单的例子即可明白此二条款(图 3.14):对

$$\gamma=(p_1\vee\overset{\text{真}}{p_2}\vee\bar{p}_3)\wedge(\bar{p}_1\vee\overset{\text{真}}{p_3})\wedge(\bar{p}_1\vee\overset{\text{真}}{p_2}\vee\bar{p}_3)\wedge(p_1\vee\overset{\text{真}}{p_2}\vee p_3)\wedge(\bar{p}_1\vee\overset{\text{真}}{\bar{p}_2}\vee\bar{p}_3)$$

有:(i) γ 可满足则 G 有 5 阶完全子图;

(ii) G 有 5 阶完全子图则 γ 可满足.

证 (i) 对于使 γ 为真的赋值,在每个子句中取出一个真文字即可;

(ii) 有 5 个顶点,两两间有边相连,既然点 1 和点 2 之间有连线,则点 1 和点 2 不在同一子句,于是这 5 个顶点出自 5 个不同的子句,即每子句出一个点.由于任二点间皆有连线可知此 5 个点相应的 5 个文字两两互不相反.于是有赋值法使 5 个文字皆为真,即使 γ 为真.即 γ 可满足. □

p_2 p_3 $\bar{p}_1$ p_2 p_2

图 3.14

注 完全子图与 01 矩阵.

图可以由 01 矩阵表示.对于 n 个点的图可按下法作一个 $n\times n$ 方阵.其元素 α_{ij} 定义为

$$\alpha_{ij}=\begin{cases}1, & \text{若 } i\neq j \text{ 但 } ij \text{ 间有边连接},\\ 0, & \text{若 } i\neq j \text{ 但 } ij \text{ 间无边连接} \quad i,j=1,2,\cdots,n.\\ 1, & \text{若 } i=j,\end{cases}$$

一个图是否有 K 阶完全子图,即是问从此图的 01 矩阵中是否可挑选出 K 个行与 K 个列,使这 K 个行的行号与这 K 个列的列号完全相同,且这些行与列的交叉点上的 K^2 个元素全为 1.

可将这个关于 01 矩阵的问题称作对称 01 方阵的 K 阶对称全 1 子方阵问题.简称作全 1 子方阵问题.

可以认为全 1 子方阵问题是最简美的 NP 完全问题,它可能会有利于对 $P=?$ NP 问题的思考.

定义 1 图中的团伙是该图的一个极大完全子图,即图中不存在一个更大的完全子图,它以该完全子图为自己的子图(图 3.15).

注 团伙即图中一个不能扩展的完全子图.

定义 2 最大团伙问题,此问题是对任一给出的图要求出最大团伙的体积即顶点数.

注意,最大团伙问题不是要回答 yes 或 no,而是要计算函数值.

命题 1 最大团伙问题是“NP 难度的问题”.即如果对此问题存在着多项式型复杂度的求解算法,则对任一属于 NP 的问题也就有了多项式型复杂度的求

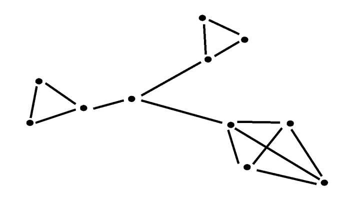


图 3.15

解算法.

证　显然,从给定图的任一给定团伙中移去任一顶点及包含该顶点的全部边,所剩仍为完全子图,于是若有快算法解决最大团伙问题则就有快算法解决完全子图问题,即最大团伙问题解决后只需问最大团伙的体积是否$\geqslant K$即解决了完全子图问题.于是最大团伙问题难于或等于一个 NP 完全问题——完全子图问题. □

定义 3　说顶点集 S 是图 $G=(V,E)$的一个顶点覆盖,是指 $S\subseteq V$ 且对每条边$(x,y)\in E$有 $x\in S\vee y\in S$.即牵制了全部边的顶点集称为顶点覆盖.

定义 4　顶点覆盖问题是任给一图 G 和一正整数 K 后,问 G 是否存在体积为 K 的顶点覆盖.

例

图 3.16 中的○点集为顶点覆盖,图 3.17 中的○点集不是顶点覆盖.

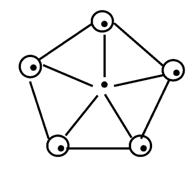


图 3.16

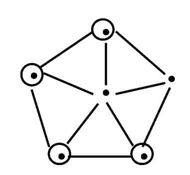


图 3.17

定理 4　设 $G=(V,E)$为一个图,$G'=(V,E')$为 G 之补图,其中 E'为不属于 E 的边(顶点偶)的全体.对 $S\subseteq V$ 记 $V-S=S'$,则有:S 为 G 中完全子图之顶点集$\Leftrightarrow S'$为 G'之顶点覆盖.

证　($\Rightarrow$)例证,图 3.18 中黑点集为 V,黑边集为 E,虚边集为 E',每两点间或用黑边连或用虚边连,S 为当中 4 黑点,有些虚边未画出.

设$\overline{ab}$为虚边,则点 a 点 b 必不能均属于 S(因若都$\in S$,则由于 S 是 G 中完全图之顶点集,必有$\overline{ab}$为黑边,矛盾)于是点 a 或点 b 属于 S',即 S'中点牵制全部虚边,即 S'为 G'之顶点覆盖.

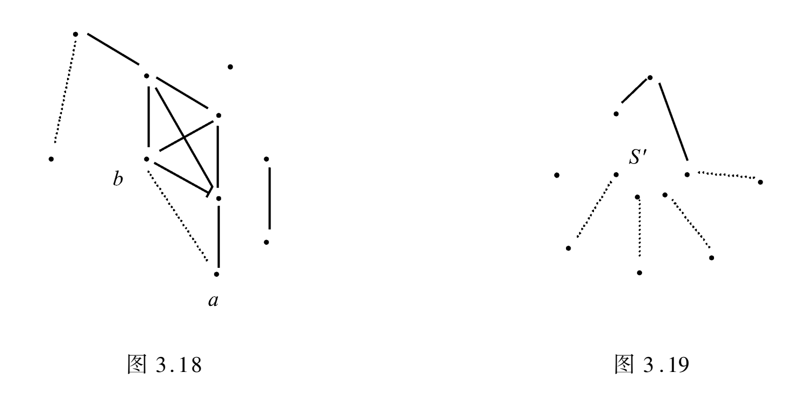


图 3.18　　　　图 3.19

(⇐)例证,图 3.19 中黑点集为 V,黑边集为 E,虚边集为 E',S' 为当中 4 黑点,有些实边未画出.

S' 点集牵动了全部虚边,设点 $a\in S$,点 $b\in S$,则边 $\overline{ab}$不被 S' 牵动,所以边 $\overline{ab}$不为虚边,为黑边,即 S 为 G 中完全子图之顶点集. □

推论 2　顶点覆盖问题是 NP 完全的.

证　设给出图 G 有 N 个顶点,则据定理 4,图 G 有体积为 k 的顶点覆盖⇔图 G' 有顶点数为 $N-k$ 的完全子图. □

推论 3　集合覆盖问题是 NP 完全的.

集合覆盖问题:任给有穷集类 $\Delta=\{S_1,S_2,\cdots,S_n\}$,任给正整数 k,问是否存在一个 k 阶子类可盖住Δ.即是否存在

$$\{S_{m_1},\cdots,S_{m_k}\},$$

使得

$$S_1\cup\cdots\cup S_n=S_{m_1}\cup\cdots\cup S_{m_k}.$$

证　一、问题∈NP.

可用如下不确定算法解决此问题.

S_1,	S_2,	$\cdots$,	S_n
要	要		要
否	否		否
?	?		?

然后问被接纳的是否 k 个集,再问这 k 个集是否已覆盖了Δ. □

二、问题为 NP 难的.

我们证明如果集合覆盖问题有快算法,则图的顶点覆盖问题也就有了快的

算法.

设图 $G=(V,E)$已被给定,共有 n 个顶点,令

$$\begin{cases} S_i = \text{以第 } i \text{ 个顶点为一端的边之全体(即第 } i \text{ 个顶点粘住的边的全体)}, \\ \Delta = \{S_1, S_2, \cdots, S_n\}. \end{cases}$$

显然,$\{S_{i_1}, S_{i_2}, \cdots, S_{i_k}\}$ 是 Δ 的集合覆盖$\Leftrightarrow\{v_{i_1}, v_{i_2}, \cdots, v_{i_k}\}$ 是图 G 的顶点覆盖.即 Δ 有一个 k 阶子类可覆盖住自己$\Leftrightarrow$图 G 有 k 阶的顶点覆盖.即图的顶点覆盖问题可以快速地归结为集合覆盖问题. □

第四章　现实生活中的 NP 难度问题及其现实处理方法——处理 NP 难度问题的拟物拟人途径

任何一个问题,如果对其求解的困难程度大于或等于求解 SAT 问题的困难程度,则称此问题为具有 NP 难度的问题,简称 NP 难度问题.

也有人认为对于 NP 难度问题应作如下定义:

任何问题如果对其求解困难到不存在多项式时间复杂度的确定算法,则称其为 NP 难度问题.

由于 P=？NP 问题没有被研究清楚,当前国际学术界还不能确定以上二定义中哪一个的外延更宽广.笔者认为用哪一个定义好并不重要,不妨定义 NP 难度问题为符合以上二定义中之任何一个定义的问题.即使用外延宽广的宽松定义.

当今时代,在纯粹科学研究,通信、交通运输、工程设计和企事业管理部门,在社会的军事、政治和商业的斗争中涌现出大量 NP 难度的问题.若按经典的纯粹数学家们所熟悉的穷举方法来求解,则计算时间动辄达到天文数字,根本没有实用价值.学术界许多有经验的人认为对于这些问题根本上就不存在完整、精确而又不是太慢的求解算法.

于是人们寄希望于非完整的启发式算法.这种算法对于很刁钻古怪的例子可能会失败,但在通常面临的实际情况下却能算得既精确又相当地快.

到哪里去寻求启发？这是一个根本的难点.中国古代的画论有“外师造化内得心源”的说法,即认为智慧应源于大自然与自身的心灵.事实上,自然界中的各种现象以及人类共同生活中的社会经验,尤其是处理相互关系中许多矛盾的各种公关手腕都是很好的源泉.许多数学中的概念方法与策略事实上都是来源于人类对大自然的观察,来源于他们在社会中生存奋斗的社会经验.最明显的例子是纯粹数学数理逻辑的专门领域递归论.其中一个极为重要的现代方法——有穷损害优先方法,就十分明显地露出了人类的公关手腕或社会经验的痕迹.许多优秀的数学家在书写论文或作讲演时都不愿谈他们的思想的来源.更有甚者,他们往往竭力掩盖其朴素的自然的或社会的源头.而给人以与生活无关、十分高雅的印象.这种情况,当然在某种程度上是因为老练的数学家们要保护自己辛苦得来的知识产权,并使其作品具有玄妙的形象以得到尽可能高的社会评价.但是另一方面广大的社会公众与政府当局也有责任.在聆听一种数学理论的时候,他们往往认为那种令人似懂非懂、十分遥远的理论为深奥为高雅.而对于自己能够

明了能够把握的理论就评价不高,更无崇拜之情.这在某种程度上又迫使了学者们故弄玄虚不把真实的思想源头谈出来.当然,公众的这种情绪亦属可以谅解,这种情绪的产生一方面是自卑所致,另一方面是由于学者们的误导的影响.至于还有大量的不成熟的学者,他们对学问本身尚无亲切的体会,他们的讲演和作品基本上是在不十分忠实地重述前人的著作,他们不会谈出学术的朴素思想源头,那更是十分自然的事情了.

这种学者与公众的互相作用,使得学术的表叙愈来愈虚伪,实在是妨碍了学术的健康发展,应该返朴归真才好.

不过以上分析也可以使我们相信,拟物拟人的途径,本来就是科学的正统,而不是邪门歪道.只不过自 19 世纪末至 20 世纪中以来,为打好数学科学的严格基础,公理化符号化的思潮风起云涌,使人们暂时淡忘了这种科学的正统而已.随着新的有意义的困难问题例如 NP 难度问题的涌现,公理化符号化的方法会逐渐显出自己的不足,朴素的拟物拟人的途径会逐渐被人们所选用.

特别值得指出的是,对于所得的启发式算法,在遇到刁钻的实例而表现出疲软失败的时候,人们可以十分自然地通过分析算法在该次失败中的表现而对它有针对性地加以改善和强化.经过若干次自然的改善和强化之后,一个从实用角度看来叫人很是满意的算法就会呈现在我们的面前.对此笔者已有一些经验.

拟物方法是一个许多人会感到有趣有用的方法.其工作路径是,到物理世界中去寻找出与原始数学问题等价的自然现象,然后观察其中物质运动的演化规律,从中受到启发以得出形式化的、对于数学问题的求解算法.单纯的拟物方法就已能解决许多问题.在遇到刁钻的问题时还可将拟物方法与拟人方法联合使用形成所谓拟物拟人方法.对其工作的方式可作如下解释、描叙和评论.

由于物理状态的演化天然地是按照使其 Lagrange 函数的时间积分达到最小值的方式进行,这就决定了拟物算法最终在数学上落实为优化问题.然而用数学方法求解优化问题,常常会碰到计算落入局部最小值陷阱的困难境地,对于如何跳出局部最小值陷阱,让计算走向前景更好的区域中去的问题,拟物方法已无能为力.但是,人类在最近几千年的共同生活中形成了丰富的社会经验,利用这些经验往往可以启发出好的“跳出陷阱”的策略.我们可以将这种把人类的社会经验形式化为算法用以求解某些特殊困难的数学问题的方式称为拟人途径.

拟物拟人算法的效率通常比生物遗传算法、神经网络方法、淬火方法要高,其原因是它有针对性地为具体问题找到了非常贴切的物理世界,而不是像在遗传、神经网络、淬火方法中那样依赖于一个惟一的始终不变的因而往往是不太贴切的物理体系.另外,拟人方法是向人学习,而人比起遗传、神经网络、淬火世界里的那些蛋白质单个的神经元及晶体显然有高得多的智慧.当然,这里的关键在于合适的数学形式化前夜的艺术和手段,要得到它们也不是一件很容易的事情,

需要长期艰苦细心的工作.这种得出算法的过程的艰苦性,可以说是拟物拟人途径的一个缺点.

另外,生物遗传算法、神经网络法、淬火法虽然针对性较差,但其适应性较强,并且亦有其深刻的思想根源和自然背景.我们曾尝试将拟物拟人法与这些方法结合起来,结果发现,在肯动脑筋并且机遇好的时候,二者能结合得很好,能产生新的效能更高的算法.

至于纯粹的拟人方法,其途径是将人类在最近几千年的生存斗争中所形成的某些经验某些作法说完整说清楚,然后加以抽象化形式化,最后形成算法以求解 NP 难问题.

此方法的关键难点在于为给定的 NP 难问题找到相应的有悠久历史的人类活动.但是一旦找到,必然能很顺利地发展为高效能的求解算法.

我们举出了六个例子,用以显示如何从这种拟物拟人的角度来处理现实生活中的 NP 难度问题.可以想象,各人都有自己的聪明和由观察思考得来的经验,可产生出各种各样的求解 NP 难问题的拟物算法、拟人算法以及拟物拟人算法来.

对于 NP 难问题的现实求解,对于求解 NP 难问题的拟物拟人途径,我们已有二十余年的初步实践经验.三十年来,世界各国先进学者们在纯粹理论方面进行过严格的思考,对于某些现实生活中出现的 NP 难问题认真地进行了大量的求解计算.这些思考的结果和实际计算的经验启发了我们,使我们对科学的严格性,对纯科学的能力,对科学和技术的差别有了进一步的认识.

对于 NP 难问题的现实求解,已经发现有些近似的求解算法,它们具有良好的实用性能.可是对于这些性能的严格证明却十分困难.典型的例子是 Packing 问题.著名的希尔伯特第 18 问题,即开普勒问题,事实上是一个蜕化了的简单的 Packing 问题.它是要研究密布于整个无穷的三维欧氏空间中的无穷个相同大小的刚性实心球所可能形成的最紧密的布局图案.最后应求出这些刚性实心球所可能达到的对于三维空间的最大占有密度并加以严格的证明.

高斯猜测,球心构成正四面体的顶点的四个两两相切的等球即是某种最紧密布局的一个图案周期.它对空间的占有密度为$\sqrt{2}\pi/6\approx0.74048$.

其实,任何一个卖过苹果的小贩都是十分熟悉这个图案的,他们实际上就常常依照这个图案摆放他们的苹果.开普勒问题看来十分简单,事实上大多数的初中毕业学生都能猜想出其答案来,可是历经数百年至今仍无得到世界公认的严格证明被人作出.

可以想象,关于日常生活经常出现的 NP 难度问题——有限形式的 Packing 问题,其难度比开普勒问题要大得多,要想对这些问题的有用算法的效能作出严格的证明,还不知需耗费全人类多少个世纪的精力.也许人类对于这种效能永远

也作不出一个我们今天的美学理想所需求的传统数学式的严格证明来.

因此,有人认为,对于这些实用的算法,绝对严格的算法分析是既做不到又无必要.于是近年来国际学术界出现了经典数学式算法分析的代用品——Benchmark(算法试金石).即由有一定威望和水准的研究机构或学者个人出面,针对某一个有名的 NP 难问题,编制出若干有代表性的具体实例——Benchmark.实例中的每一个参数都被确切地给出.对求解算法的考核办法是,该算法对这些实例一一进行求解计算,最后记下其解答的精度和计算时间.如果按此算法进行计算的精度、速度的综合性能好则说此算法好,反之则此算法不好.

笔者认为这种 Benchmark 式的评价标准虽然目前还不够十分完善,但终究会被大多数人所接受.它既有充分的科学依据又有其深厚的现实根源,不可轻视.有志于求解 NP 难问题的人可以利用 Benchmark 进行踏实的计算实验,而不必躲避它们.这样作会有利于改进老的算法,得到高性能的新算法,避免使研究工作陷入困境,长期地对低性能的算法去作劳而无功的分析,浪费众多优秀学者的精力和时间.

§1. 求解 Packing 问题的拟物方法

考虑如下问题:在一个已知的容器中希望能放下 N 个已知不同形状大小的物体,其中界限容器的封闭边境以及各个物体都是不可入的刚性实体,如果客观上放不下,我们要求作出放不下的判断;如果客观上放得下,则要求给出每个物体的位置和方向.

这就是所谓 Packing 问题.

我们将这 N 个物体想象为光滑的弹性实体.将容器想象为充满整个三维空间的光滑弹性物,不过其中因挖去部分实体而形成一个空腔,此空腔的大小形状相同于容器所境界的空间部分.

想象这 N 个弹性体挤缩在这个弹性空腔中.如果我们原始的刚性置入问题客观上有解,那么,这个存在挤压的弹性物体与空腔所构成的体系就会在弹性力的作用之下发生一系列的运动,最终有可能使得各个物体与空腔都恢复自己的大小与形状,因而使得置入问题的定解条件得到满足.

因此,如果我们能用数学的方法在某种意义上模拟弹性力作用之下的物体与空腔的运动方式,则我们在事实上就得到了一种置入问题的解法.

将固结于容器之上的空间笛卡儿坐标系取作绝对坐标系.对每一个物体都可以按某种方式事先指定一个固结于其上的笛卡儿坐标系,此坐标系的原点选择在物体的几何重心上.第 $i(i=1,2,\cdots,N)$ 个物体在容器中的状态由以下六个实数所描述:

$$x_i, y_i, z_i, \theta_i, \varphi_i, \Psi_i. \tag{1.1}$$

其中 x_i, y_i, z_i 表示第 i 个物体的几何重心在绝对坐标系之下的坐标;$\theta_i, \varphi_i, \Psi_i$ 表示第 i 个物体在绝对坐标系之下的欧拉角,它们表现了此物体所处的方向.

因此,整个物体容器体系的一个状态由如下 $6N$ 个实数所描述:

$$x_1, y_1, z_1, \theta_1, \varphi_1, \Psi_1, \cdots, x_N, y_N, z_N, \theta_N, \varphi_N, \Psi_N. \tag{1.2}$$

我们将物体容器体系的所有状态所构成的集合称作问题的状态空间,对于体系的任意一个确定的状态我们称为状态空间中的一个点.

我们引进二物体间的距离的概念. i, j 二物体间的距离记作 L_{ij},其中 $i \neq j$, $i, j = 1, 2, \cdots, N$. 当 i, j 二物体的交的 Lebesgue 测度大于零时,我们定义 L_{ij} 为某个负数,其绝对值等于 i, j 二物体沿其连心线方向以平移方式互相远离至交的 Lebesgue 测度为零时所经过的长度;当 i、j 二物体的交的 Lebesgue 测度为零时我们定义 L_{ij} 为某个非负数,其数值等于 i, j 二物体沿其连心线方向以平移方式互相接近至交的 Lebesgue 测度为 0^+ 时所经过的长度(图 4.1).

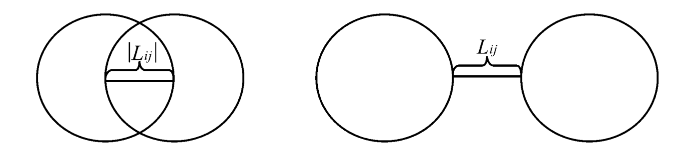


图 4.1

我们将容器外部看作带有一个空腔的弹性实体,将此实体看作第 0 个物体. 对于第 0 个物体与第 j 个($j = 1, 2, \cdots, N$)物体的距离 L_{0j} 我们作如下定义:当 $0, j$ 二物体的交的 Lebesgue 测度大于零时, L_{0j} 为某个负数,其绝对值等于 $0, j$ 二物体沿空腔的几何重心与 j 物体的几何重心的连线方向以平移方式互相远离至交的 Lebesgue 测度为零时所经过的长度;当 $0, j$ 二物体的交的 Lebesgue 测度为 0 时, L_{0j} 为某个非负数,其数值等于 $0, j$ 二物体沿空腔的几何重心与 j 物体的几何重心的连线方向以平移方式互相接近至交的 Lebesgue 测度为 0^+ 时所经过的长度(图 4.2).

我们用下式定义 i, j($i \neq j$, i、$j = 0, 1, \cdots, N$)二物体间的弹性势能 U_{ij}:

$$U_{ij} = \begin{cases} 0, & \text{当 } L_{ij} \geqslant 0 \text{ 时}, \\ L_{ij}^2, & \text{当 } L_{ij} < 0 \text{ 时}. \end{cases} \tag{1.3}$$

此式的弹性力学解释如下:当 $L_{ij} < 0$ 时 $|L_{ij}|$ 表征了 i, j 二物体挤压变形的尺

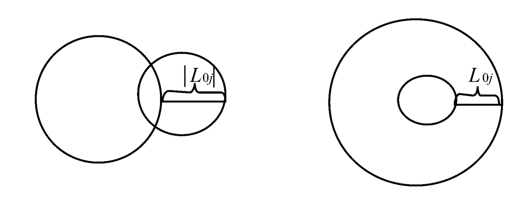


图 4.2

度,而弹性变形的势能则是正比于变形尺度的平方;当 $L_{ij} \geqslant 0$ 时, i, j 二物体的交的测度为 0,没有发生挤压,因而弹性变形势能为 0.

由下式定义 N 个物体与空腔所构成的系统的弹性势能 U:

$$U = \sum_{i=0}^{N-1} \sum_{j=i+1}^{N} U_{ij}, \tag{1.4}$$

即 $N+1$ 个物体所构成的体系的弹性势能为其中每两个物体之间的弹性势能的总和.

显然, U 是系统的状态的函数,即

$$U = U(x_1, y_1, z_1, \theta_1, \varphi_1, \Psi_1, \cdots, x_N, y_N, z_N, \theta_N, \varphi_N, \Psi_N). \tag{1.5}$$

由函数(1.5)的构造可知:(i) U 恒非负;(ii) $U > 0$ 时表明状态(1.2)不满足 N 个物体在容器中放下的要求;(iii) $U = 0$ 则表明状态(1.2)满足 N 个物体在容器中放下的要求.我们将满足 $U = 0$ 的状态称为状态空间中的可行点.

将 U 作为状态空间中的指标函数,我们按柯西的求实超越方程组的解的下降算法,即梯度法,利用电子计算机寻求状态空间中的可行点.在可行点求出后,置入问题即告解决,因为数组(1.2)即指出了使 N 个物体能放进容器中去的具体的姿态.

如果在计算相当长的时间以后还算不出可行点来,则可能是此问题无解,或者是由于以下三个困难条件的存在致使此方法失灵:(i)空间紧张;(ii)物体大小悬殊;(iii)物体个数多.

我们用(1.4)、(1.3)式定义指标函数 U,而不用从罚函数的角度看来更为自然的二点集的嵌入深度,是因为那样作没有物理意义,将来实算起来会出现一些很不协调的现象.例如局部极小点增多,在可行点附近梯度不趋于零,其模的大小失去搜索导向指标的涵义.

以上求可行点的过程实际上是对势能函数 U 求全局最小值点的过程.即我们事实上是将 Packing 问题化成了一个天然的全局优化的问题,并用优化方法来求解.

在一些具体的问题中,对计算过程的图形屏幕显示表明,各物体在容器中迭代移动的图像确实与现实物理世界中的运动情景大体一致.

§2. 求解覆盖(Covering)问题的拟物方法

所谓覆盖问题即是已知平面上的有穷个点(不动),用我们手中拥有的一定个数的半径为 R 的圆盘,如何布置圆盘的位置,才能将这些点都覆盖住.这类问题除了有理论意义外,近来又有了较明显的实际意义.例如,如何布置机器人的位置,使得已知位置的诸服务对象能得到充分的服务的问题,即是一个覆盖问题.其中圆盘的半径 R 代表机器人的手臂长度.

将平面想象为处于水平位置并绝对光滑,将这有穷个点想象为嵌死在此平面上的具有单位质量的质点.将各个圆盘想象为绝对光滑,具有单位质量且质量均匀分布在圆周上.

若随机地将这些圆盘放好,坐落在此光滑平面上,则由于质点与圆盘间的万有引力的缘故,每一个质点都会吸引圆盘来覆盖自己,从而使覆盖问题得到解决.

但为了使圆盘不是在应有尽有的情况下也能解决覆盖问题,我们应当节约使用圆盘.节约的策略当是通知每个质点,在已经被覆盖的情况下,即要求已被满足时,就再不要吸引任何圆盘来覆盖自己,以将机会即资源让给别的质点.这种策略的精神,与数理逻辑现代递归论中的有穷损害优先方法是一致的.然而它也有其物理背景,那就是屏蔽现象.

根据万有引力定理,忽略某些纯力学的细节后进行积分就可得出由一质点与一圆盘构成的简单体系的引力势函数:

$$u = \begin{cases} \dfrac{1}{R} - \dfrac{1}{r}, & \text{当 } r \geqslant R \text{ 时}, \\ 0, & \text{当 } r < R \text{ 时}, \end{cases} \tag{2.1}$$

其中 R 为圆盘的半径,r 为质点到盘心的距离.

现考虑由 M 个质点与 m 个圆盘构成的体系,其势能函数的表达式为

$$U = \sum_{i} \sum_{j=1}^{m} u_{ij}, \tag{2.2}$$

其中 u_{ij} 表示由第 i 个质点与第 j 个圆盘构成的简单体系的势能,其算式按(2.1);$\sum_{i}$ 表示对质点的求和只计入未被覆盖的点,而不计入已被覆盖住的点.

由(2.2)式的构造可知:(i) U 恒非负;(ii) $U>0$ 表明由诸圆盘的圆心坐标

所决定的系统状态 $x_1, y_1, \cdots, x_m, y_m$ 不满足 m 个圆盘覆盖住诸点的要求；(iii) $U = 0$则表明状态 $x_1, y_1, \cdots, x_m, y_m$ 满足 m 个圆盘覆盖住诸点的要求.我们将满足 $U = 0$ 的状态称为状态空间中的可行点.

将 U 作为状态空间中的指标函数,我们按柯西的求实超越方程组的解的下降算法,即梯度法,利用电子计算机寻求状态空间中的可行点.在可行点求出后,覆盖问题即告解决.数组 $x_1, y_1, \cdots, x_m, y_m$ 即指出了诸圆盘的安置方法.

如果在计算了相当长的时间以后还算不出可行点来,则可能是此问题无解,或者是由于以下三个困难条件的存在致使此方法失灵:(i)盘子个数不充裕;(ii)点多且分布宽广;(iii)点的分布绵延不断且方式古怪.

我们用(2.1)、(2.2)式定义指标函数,而不用从罚函数的角度看来更为简便自然的其他测度,是因为吸引比惩罚更自然更有效.

不太好想象的是,为什么圆盘与圆盘之间没有万有引力,以及在同一个平面上为什么圆盘与圆盘之间可以自由地互相嵌入而不产生阻力.后者是与物质实体的不可入性相冲突的.这两个不好想象的方面,正是为什么在得出 Packing 问题的拟物算法之后的近十年的时间里,我们一直未能得出覆盖问题的拟物算法的原因.

其实,自然界里会存在有两种不同物质(例如铁与磁铁),它们同类之间的作用力很小,而异类之间的作用力则相当地大.我们可想象点与圆盘分别由这两种不同的物质组成.

另外,关于实体的不可入问题,可以用一个初看起来有点牵强附会的想象予以解决.将原始的平面扩充,想象为 $m + 1$ 个间隔非常小的平行平面, M 个质点都嵌在第 0 个平面上,而 m 个不同的圆盘则分别放在其余的 m 个不同的平面上.这样,盘子在运动中不存在相互嵌入的问题,而其运动图案在第 0 个平面上的投影则正是我们所需要的一个平面上的可自由互相嵌入的诸圆盘在诸不动点的吸引之下进行运动所形成的图案.

此方法的精神在于,在某些困难的算法问题面前,当纯粹数学的思维显得似乎枯竭的时候,我们可到自然界中去寻求启发,而在最后设计算法时又不必拘泥于自然现象的绝对真实性.至于如何作适当的修正或想象,则以有利于原始的算法问题的解决为准.

§3. 求解 SAT 问题的拟物方法

考虑如下的合取范式(CNF),问它是否可满足:

$$[\neg P_1 \vee \neg P_2] \wedge [P_1 \vee P_2], \tag{3.1}$$

其中 P_1, P_2 为独立命题变元, $\urcorner P_v$ 表示 P_v 的非,每个命题变元的取值范围为 $\{0,1\}$,数 0 和 1 的逻辑意义分别为假和真,这是一个具体的 SAT 问题.

考虑如下变换,它将二维 Euclid 空间 $R^2=\{-\infty,+\infty\}^2$ 中的任意一点 (x_1,x_2) 变换成二元 Boole 空间 $\{0,1\}$ 中的点 (P_1,P_2).

$$P_i=\begin{cases}1, & 若\ x_i\geqslant 1;\\ 0, & 若\ x_i\leqslant 0;\\ 任意地给定 0 或 1, & 若\ x_i\in 开区间(0,1)。\end{cases}\qquad(3.2)$$

因为 $P_1\vee P_2$ 为真意味着 $P_1=1$ 或 $P_2=1$ 即 $x_1\geqslant 1$ 或 $x_2\geqslant 1$; $\overline{P_1}\vee\overline{P_2}$ 为真意味着 $P_1=0$ 或 $P_2=0$,即 $x_1\leqslant 0$ 或 $x_2\leqslant 0$.于是,变换(3.2)式将 CNF(3.1)的可满足性问题等价地变换为如下 R^2 中的覆盖问题:在图 4.3 所示的平面上是否存在一个点,它既处于斜阴影区域又处于垂直阴影区域?换言之,对于已知的 2 个区域而言,平面上是否存在着一个点,它被其中的每一个所覆盖?因此,可将 CNF 可满足性问题看成覆盖问题,只不过在经典的覆盖问题中是点不动区域动,而现在则是区域不动点动.

将直阴影区域与斜阴影区域都看成带负电荷的钢板,它们之间被电介质薄板隔开.将 2 块钢板压紧平放好,上面再放 1 块玻璃板.此时如果在玻璃板的任一给定位置上放 1 颗带正电荷的玻璃珠,则 2 个带电钢板中的每个都用自己的电场力独立作用于玻璃珠以将它引向自身.玻璃珠一旦进入某钢板后此钢板由于屏蔽现象的原因即对其失去引力,待到又离开这块钢板了,则此板又对珠子恢复引力,再次将其引向自身.除非玻璃珠达到 2 块钢板的公共部分,它将一直处于不停的运动之中.这样,在 2 块钢板诱导出的 2 个电场力的共同作用下,玻璃珠最终会逐渐被引至由这 2 个区域共同覆盖的公共部分.于是覆盖问题得到解决.按变换(3.2)式,原始的可满足性问题得到解决.

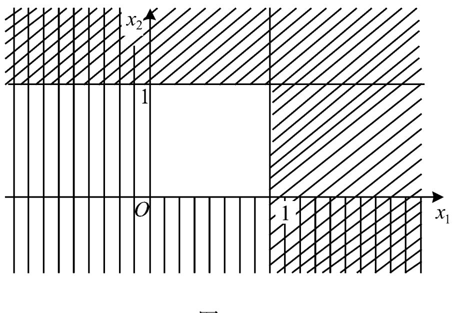


图 4.3

玻璃珠所受的力为

$$F=-\operatorname{grad}[U_1(x_1,x_2)+U_2(x_1,x_2)],\qquad(3.3)$$

其中 $U_\nu(x_1,x_2)$ 为第 ν 块钢板诱导出的电场的势能.按照物理意义, $U_1(x_1,x_2)$ (直阴影钢板诱导出的势能)在第 1 象限外部(导体上)值为零,而在内部(真

空)则满足 Laplace 方程.在针对第 1 象限利用分离变量法求解偏微分方程边值问题后,得出 $U_1(x_1, x_2)$在整个平面上的分布为

$$U_1(x_1, x_2) = \begin{cases} x_1 x_2, & \text{若 } x_1 \geqslant 0 \wedge x_2 \geqslant 0, \\ 0, & \text{否则}. \end{cases} \tag{3.4}$$

同理,相应于斜阴影钢板,可得

$$U_2(x_1, x_2) = \begin{cases} (1 - x_1)(1 - x_2), & \text{若 } x_1 \leqslant 1 \wedge x_2 \leqslant 1, \\ 0, & \text{否则}. \end{cases} \tag{3.5}$$

总势能为

$$U(x_1, x_2) = U_1(x_1, x_2) + U_2(x_1, x_2). \tag{3.6}$$

显然, $U(x_1, x_2)$为非负函数,并且 $U(x_1, x_2) = 0$ 等价于点(x_1, x_2)是与 CNF-SAT 问题(3.1)相应的覆盖问题的解.因此,我们的 CNF-SAT 问题被转化成了总势能函数 $U(x_1, x_2)$的求最小值点问题——优化问题.设 $U(x_1^*, x_2^*)$为最小值点,若 $U(x_1^*, x_2^*) = 0$,则按(3.2)式即得出原始 SAT 问题的解(P_1^*, P_2^*);若 $U(x_1^*, x_2^*) > 0$,则原始 SAT 问题无解.

对于一般的 CNF

$$\bigwedge_{i=1}^{l} \left[P_{i,1} \vee P_{i,2} \vee \cdots \vee P_{i,k_i} \vee \neg P_{ri,1} \vee \neg P_{ri,2} \vee \cdots \vee \neg P_{ri,K_{ri}} \right], \tag{3.7}$$

其中 $P_{i,1}, \cdots, P_{i,k_i}, P_{ri,1}, \cdots, P_{ri,K_{ri}}$ 为命题变元集$\{P_1, \cdots, P_m\}$中两两不同的命题变元,相应地写出总势能函数

$$U[x_1, x_2, \cdots, x_m] = \sum_{i=1}^{l} U_i[x_1, x_2, \cdots, x_m], \tag{3.8}$$

其中

$$U_i[x_1, x_2, \cdots, x_m] = \begin{cases} [1 - x_{i,1}][1 - x_{i,2}]\cdots[1 - x_{i,k_i}] x_{ri,1} x_{ri,2} \cdots x_{ri,K_{ri}}, \\ \qquad \text{若 } x_{i,1} \leqslant 1 \wedge \cdots \wedge x_{i,k_i} \leqslant 1 \wedge x_{ri,1} \geqslant 0 \wedge \cdots \wedge x_{ri,K_{ri}} \geqslant 0, \\ 0, \qquad \text{否则}, \end{cases} \tag{3.9}$$

这里 $U_i(x_1, x_2, \cdots, x_m)$的物理意义为 m 维 Euclid 空间中带负电的导体

$$\mathrm{R}^m - [x_{i,1} \leqslant 1 \wedge \cdots \wedge x_{i,k_i} \leqslant 1 \wedge x_{ri,1} \geqslant 0 \wedge \cdots \wedge x_{ri,K_{ri}} \geqslant 0]$$

所诱导出的电场的静电势能,其中 $x_{i,1},\cdots,x_{i,k_i},x_{ri,1},\cdots,x_{ri,K_{ri}}$ 为实变元集 $\{x_1,\cdots,x_m\}$ 中两两不同的实变元.

CNF(3.7)的 SAT 问题等价于对(3.8)式中总势能函数 $U(x_1,x_2,\cdots,x_m)$ 求最小值点的问题.设 $x_1^*,x_2^*,\cdots,x_m^*$ 为最小值点,若 $U(x_1^*,x_2^*,\cdots,x_m^*)>0$,则 SAT 问题无解,若 $U(x_1^*,x_2^*,\cdots,x_m^*)=0$,则依变换

$$P_v^*=\begin{cases}1, & \text{若 } x_v^*\geqslant 1,\\ 0, & \text{若 } x_v^*\leqslant 0, \quad \text{对于 } v=1,2,\cdots,m,\\ \text{任意地给定 0 或 1}, & \text{若 } x_v^*\in(0,1),\end{cases} \tag{3.10}$$

$P_1^*,P_2^*,\cdots,P_m^*$ 为 CNF(3.7)的成真指派.因此原始 SAT 问题得以解决.因此,根据函数 U,SAT 问题可望用柯西梯度法加以解决.

这种依柯西梯度法求总势能函数 $U(x_1,x_2,\cdots,x_m)$ 的最小值点的迭代过程,事实上吻合于物理世界中那个带正电荷的质点走向最低势能位置的运动过程.

注 由表达式(3.9)可见,我们的 $U_i(x_1,x_2,\cdots,x_m)$ 在广大的地区为零,与先前学者所提出的指标函数不同.由于玻璃珠频繁地在 Euclid 空间中单位立方体$[0,1]^m$ 的边界上运动,而边界上 U 的梯度是取决于函数在两侧的分布,而不是单方面地取决于一侧的分布的,因此,即使在$[0,1]^m$ 中讨论问题,我们的函数 $U_i(x_1,x_2,\cdots,x_m)$ 也是不同的.至于考虑到 U_i 的物理来源,则这种不同就更是深刻了.

§4. 求解不等圆 Packing 问题的拟物拟人方法

本节利用不等圆 Packing 问题的求解来显示拟物拟人方法的思想和技术.

4.1 问题的提法

假设已知一个圆形的空盒子(容器),另外又已知一些不同的圆饼,问能否将这些圆饼互不重叠地放进空盒子中去(图 4.4).此问题更形式的表达如下:

对于任意给定的正整数 M 和任意给定的 $M+1$ 个正实数 $R_0,R_1,\cdots,R_M$,是否存在 $2M$ 个实数 $x_1,y_1,\cdots,x_M,y_M$,使得对于$\{1,2,\cdots,M\}$ 中任意的正整数 i 有

$$\sqrt{x_i^2+y_i^2}\leqslant R_0-R_i, \tag{4.1}$$

对于$\{1,2,\cdots,M\}$ 中任意两个不同的正整数 i,j 有

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geqslant R_i + R_j . \qquad (4.2)$$

如果存在,则请具体给出一组合乎条件的实数.

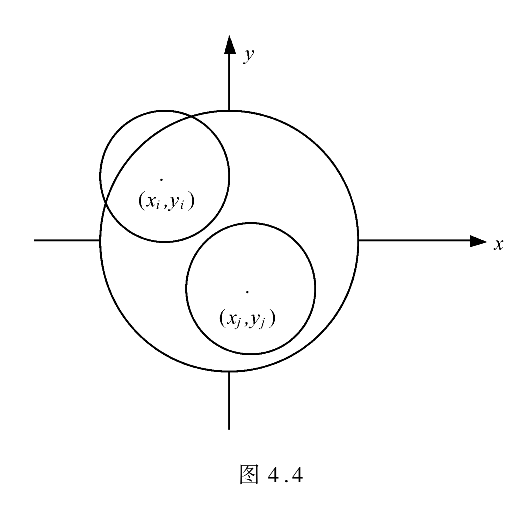


图 4.4

4.2　拟物与拟人的思路

4.2.1　拟物的思路

我们在二维 Euclid 空间中考虑问题.将这 M 个圆饼都想象为光滑的弹性实体;将这个容器想象为整个的二维无穷弹性实体,在因挖去了半径为 R_0 的一个圆饼之后而剩下的无穷部分.

设想这 M 个圆饼被挤缩在这个容器中.由于各个物体都有要恢复自己形状大小的趋势,它们之间产生了挤压弹性力的相互作用.在这种挤压弹性力的驱使之下,各物体会产生一系列的运动.作为这种运动的结果,可能就是问题的解的确立——各物体到达某种相互合适的位置,两两互不嵌入.

如果我们能用某种数学方法将这一系列运动加以模拟,则事实上就得到了一种求解圆形 Packing 问题的算法.

4.2.2　拟物算法

将二维笛卡儿坐标的原点取在容器的中心上(见图 4.4).记第 i($i = 1, 2, \cdots, M$)个圆饼的圆心坐标为 x_i, y_i,则第 i 个圆饼与圆盘之间的嵌入深度为:

$$d_{0i} = \begin{cases} \sqrt{x_i^2 + y_i^2} + R_i - R_0, & \text{若 } \sqrt{x_i^2 + y_i^2} + R_i > R_0; \\ 0, & \text{否则}. \end{cases} \qquad (4.3)$$

第 i 与第 j 两个不同的圆饼之间的嵌入深度为

$$d_{ij} = \begin{cases} R_i + R_j - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \\ \qquad \text{若 } \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} < R_i + R_j; \\ 0, \quad \text{否则}. \end{cases} \qquad (4.4)$$

显然, d_{ij} 与 d_{0i} 恒非负. $d_{ij} > 0$ 表示第 i 与第 j 二物体之间互有嵌入, $d_{0i} > 0$ 表示第 0 与第 i 两个物体之间互有嵌入. $d_{ij} = 0$ 表示第 i 与第 j 二物体之间没有

嵌入, $d_{0i}=0$ 表示第 0 与第 i 二物体之间没有嵌入.根据弹性力学,二光滑弹性物体之间的挤压弹性势能正比于它们之间互相嵌入深度的平方.因此可以用 $d_{\lambda\mu}^2$ 来表征第λ 与第μ 个物体之间的挤压弹性势能:

$$u_{\lambda\mu} = d_{\lambda\mu}^2, \qquad \lambda, \mu = 0, 1, 2, \cdots, M, \qquad \lambda \neq \mu, \tag{4.5}$$

可以认为第 i 个圆饼具有的挤压弹性势能 U_i 为

$$U_i = \sum_{\lambda=0, \lambda \neq i}^{M} u_{\lambda i}, \qquad i = 1, 2, \cdots, M, \tag{4.6}$$

整个体系的势能为

$$U = \sum_{i=1}^{M} U_i. \tag{4.7}$$

由(4.3)~(4.7)式,整个体系的势能 U 是全部圆饼的坐标 $x_1, y_1, \cdots, x_M, y_M$ 的已知函数:

$$U = U(x_1, y_1, \cdots, x_M, y_M). \tag{4.8}$$

显然我们有:(i) $U(x_1, y_1, \cdots, x_M, y_M)$ 在整个 $(-\infty, +\infty)^{2M}$ 上有定义,非负;(ii)若 $U(x_1, y_1, \cdots, x_M, y_M) > 0$,则 $(x_1, y_1, \cdots, x_M, y_M)$ 不是圆形 Packing 问题的解;若 $U(x_1, y_1, \cdots, x_M, y_M) = 0$,则 $(x_1, y_1, \cdots, x_M, y_M)$ 是圆形 Packing 问题的解.因此,圆形 Packing 问题转化成了对于已知函数(4.8)式的优化问题,即求出函数的最小值点 $(x_1^*, y_1^*, \cdots, x_M^*, y_M^*)$,若 $U(x_1^*, y_1^*, \cdots, x_M^*, y_M^*) = 0$,则 $x_1^*, y_1^*, \cdots, x_M^*, y_M^*$ 即是问题的解,若 $U(x_1^*, y_1^*, \cdots, x_M^*, y_M^*) > 0$,则问题无解.

对此优化问题,我们有现成的求解算法,梯度法或称最陡下降法.可以指出,在按梯度法求解的过程中 $(x_1, y_1, \cdots, x_M, y_M)$ 不断地演化,这种演化过程与挤缩在容器中的诸光滑弹性圆饼的运动图像是完全一致的.

4.2.3 拟人策略的来源

拟物算法在执行过程中,当问题在客观上比较困难时,往往会碰到“卡壳”(get stuck)的情形,即计算落入了局部极小值点的陷阱.此时我们走到了势能函数 U 的局部极小值点:

$$P_{\mathrm{LM}} = (x_1, y_1, \cdots, x_M, y_M)_{\text{local minimum}},$$

由于梯度为零,现在按梯度法已不知计算如何往下进行.可是此时势能 U 仍然大于零,并且我们不能据此作出问题无解的判断,因为很有可能函数 U 在其全

局最小值点 P_{GM} 上为零(见图 4.5).

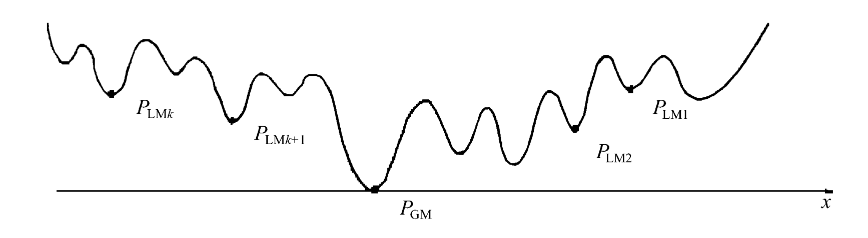


图 4.5

为解脱此种困境,纯粹的拟物算法是重新随机地选取初始值(x_1, y_1, …, x_M, y_M),然后再进行一轮新的拟物计算.这种计算虽然从原则上讲具有最终收敛的性质,但是实算的经验说明其效率是低的.此时,有前途的办法是提出好的“跳出陷阱”的策略,将计算点从局部极小值的陷阱中取出,而置入到具有更好的前景的位置上去,然后接着进行新的拟物计算.

这种“跳出陷阱”的策略可通过观察与体会人类的社会现象而得出,因而被称之为拟人策略.

在热闹拥挤的公共汽车中,受挤压最甚者总是设法改换自己的位置,而处境宽松者往往也会让出一部分空间给予别人.

将挤缩在容器内的 M 个圆饼以及容纳这 M 个圆饼的半径为 R_0 的圆盘看成一个公共汽车的车厢.于是以上对于乘车现象的观察启发我们得出以下策略.

在“卡壳”的情形,可以挑出受挤压最甚的圆饼,将它随机地放到圆盘中的某一个地方去;有时也可以挑出那个感受最宽松的圆饼,将它随机地放到圆盘中的某个地方去.挑出受挤压最甚的圆饼重新安放,是为它提供机会,希望它的痛苦有可能最终得到解脱;挑出感受最宽松的圆饼重新安放,是因为它很有可能占据了多余的空间,将它调走之后就有可能腾出这部分空间,以供那些最需要空间的受挤压的圆饼来享有.

“三十六计走为上”,“天之道损有余而补不足”,我们的策略与这些说法的精神是十分一致的,或者甚至可以说作者在潜意识中长期深刻地受到了这些说法的影响,事实上是这些说法暗示作者,使他们结合自己的经验得到了以上的策略.

我们可以将挑出受挤压最甚的圆饼的策略称之为压力解除策略,将挑出感受最宽松的圆饼的策略称之为资源让与策略.

4.3 拟人策略与求解圆形 Packing 问题的拟物拟人算法

现在我们将压力解除策略与资源让与策略综合为一个统一的拟人 Packing 策略,并对它作出确切的表述.

定义 1 在任一时刻,称这 M 个圆饼的圆心坐标 $x_1, y_1, \cdots, x_M, y_M$ 为系统在此刻的格局.

定义 2 在格局 $x_1, y_1, \cdots, x_M, y_M$ 之下,定义圆饼 i 的绝对痛苦度 DP_i 为此时其自身所具有的挤压弹性势能:

$$DP_i = U_i. \tag{4.9}$$

定义 3 在格局 $x_1, y_1, \cdots, x_M, y_M$ 之下,定义圆饼 i 的相对痛苦度 RDP_i 为此时其自身所具有的挤压弹性势能除以其半径的平方:

$$RDP_i = U_i / R_i^2. \tag{4.10}$$

拟人 Packing 策略 设按拟物算法计算第 1 次达到了局部最小值点 $P_{LM} = (x_1, y_1, \cdots, x_M, y_M)$. 我们在全体 M 个圆饼中确认出此时相对痛苦度 RDP 最大的圆饼 i. 保持其他圆饼的位置不变,将圆饼 i 的圆心位置重新随机地确定为圆盘内的某一个点. 于是计算点从局部最小值点$(x_1, y_1, \cdots, x_M, y_M)$跳至了一个新点 $x_1, y_1, \cdots, x_i, y_i, x_M, y_M$. 然后从此新点出发按拟物算法往下进行新的一轮计算.

设计算第 $k+1(k=1,2,\cdots)$次达到了局部最小值点 $P_{LM,k+1}$. 依次作如下动作:

(i) 在全体 M 个圆饼中确认出此时相对痛苦度 RDP 最大的圆饼 i.

(ii) 若第 k 次"跳出陷阱"是按相对痛苦度最大挑出的圆饼,且那个圆饼不是此圆饼 i,则将圆饼 i 的圆心位置重新随机地确定为圆盘内的某一点.

(iii) 若第 k 次"跳出陷阱"是按相对痛苦度最大挑出的圆饼,且那个圆饼就是此圆饼 i,则暂且保持格局不变,确认出此格局下绝对痛苦度最小的圆饼 j,将此圆饼 j 的圆心位置重新随机地确定为圆盘内的某一点.

(iv) 若第 k 次"跳出陷阱"是按绝对痛苦度最小挑出的圆饼,则将圆饼 i 的圆心位置重新随机地确定为圆盘内的某一点.

利用以上拟人 Packing 策略,在拟物算法的基础上,我们事实上即得到了如下求解圆形 Packing 问题的拟物拟人算法:

计算开始按拟物算法进行,直至达到某个极小点,若此时势能 U 为零则成功停机,否则按拟人 Packing 策略,将计算点跳至一个新的位置,再按拟物算法往下进行,……,直至达到某个势能 U 为零的极小点(即问题的解)为止.

4.4　对计算程序的描述

为执行拟物拟人算法,对于某一大类算例我们设计了如下计算程序:

(1) 在以原点为圆心,半径为 R_0 的圆盘内随机地给出 M 个点(x_1, y_1), $(x_2, y_2), \cdots, (x_M, y_M)$.由此确定初始格局 $X(x_1, y_1, \cdots, x_M, y_M)$.

令 $U_{old} = 0, t = 0, l_0 = 0, h = 1$;

(2) 如果 $h < 10^{-3}$,则转(5);

(3) 计算 $U(X)$;

(4) 若 $U(X) < 10^{-6}$,则成功停机;否则计算 U 的梯度向量 Grad U,

(a) 若 $U(X) < U_{old}$,则 $U_{old} \Leftarrow U(X)$, $X \Leftarrow X - (\text{Grad}\, U) \cdot h$,转(2);

(b) $U(X) \geqslant U_{old}$,则 $U_{old} \Leftarrow U(X)$, $h \Leftarrow h \cdot 0.8$, $X \Leftarrow X - (\text{Grad}\, U) \cdot h$,转(2);

(5) 确认出相对挤压弹性势能最大的圆饼 $l, 1 \leqslant l \leqslant M$;

(6) 若 $l = l_0$,则 $t \Leftarrow t + 1$;

(7) 若 $t < 1$,则在圆盘内重新随机地选点(x_l, y_l), $l_0 \Leftarrow l$, $h \Leftarrow 1$,转(2);

(8) 若 $t = 1$,则确认出绝对挤压弹性势能最小的圆饼 $l, 1 \leqslant l \leqslant M$,在圆盘内重新随机地选点$(x_l, y_l)$, $t \Leftarrow 0$, $l_0 \Leftarrow 0$, $h \Leftarrow 1$,转(2).

§5. 求解 SAT 问题的拟物拟人方法

5.1　拟人策略在求解 SAT 问题的拟物算法中的应用

人类有多种具体的社会经验可以用来加快求解 SAT 问题的拟物算法的计算过程.我们有可能详尽并严格地表述出这种配合拟物计算的拟人策略的产生过程,包括其思想和技术细节.

5.2　物理过程的离散化

在§3 求解 SAT 问题的拟物方法中,CNF(3.7)的 SAT 问题等价于对(3.8)式中总势能函数 $U(x_1, x_2, \cdots, x_m)$求最小值点的问题.

设 $x_1^*, x_2^*, \cdots, x_m^*$ 为最小值点,若 $U[x_1^*, x_2^*, \cdots, x_m^*] > 0$,则 SAT 问题无解,若 $U[x_1^*, x_2^*, x_m^*] = 0$,则依变换(3.10), $P_1^*, P_2^*, \cdots, P_m^*$ 为 CNF(3.7)的成真指派,反之若 $P_1^*, P_2^*, \cdots, P_m^*$ 为 CNF(3.7)的成真指派,则总势能函数

$$U[P_1^*, P_2^*, \cdots, P_m^*] = 0. \tag{5.1}$$

因此求实函数 U 的零点问题,可以只限于在离散的$\{0,1\}^m$ 空间里进行,即可以

离散化.

5.3 对总势能函数寻优搜索的大致框架

由于对实数的运算比对 01 数的运算要慢许多,真正模拟实 m 维 Euclid 空间 R^m 中带电粒子运动的模拟算法,在目前国际学术界所涉及的低维情形(如小于 10^5 维)不会显示出其真正的优点.而由于(5.1)式的保证,我们可以只对总势能函数 $U(x_1, x_2, \cdots, x_m)$ 在 R^m 中的子集,离散的$\{0,1\}^m$ 空间上进行寻优搜索.注意,即使在这种离散的搜索中,我们的连续的物理模型所带给我们对 SAT 问题的感觉与理解,其价值仍然是非常之高的.因为利用这种感觉与理解能启发出好的搜索策略.与这种策略有关的现象在本质上与离散或连续没有关系,但在连续的表达方式之下,特别在有了物理意义之后,人们能看得更为直观与清晰.另外,这种连续模型使我们能很自然地将离散计算嵌入到连续计算中去,从而可以利用连续数学中强有力的工具,从理论上得出在离散搜索中对于一个点及其邻域中全部点上的总势能函数值的快速计算.

定义 1 把对 $x_1, x_2, \cdots, x_m$ 的一组真值指派看成$\{0,1\}^m$ 空间中的一个点 X,将与点 X 恰在某一个变元上取值不同的点的集合称为 X 的邻域.

显然,任一点的邻域中恰有 m 个点.对于点 X',若 $U(X') < U(X)$,则称 X' 为 X 的邻域中的下降点.若下降点 X' 是由 X 翻转变元 X_v 的值而来,则称 x_v 为 X 的下降变元.

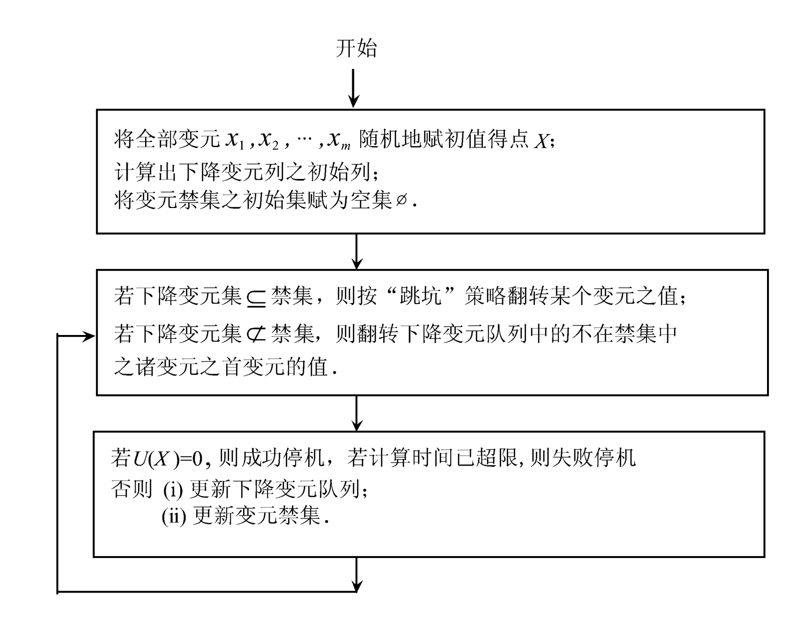
我们的寻优搜索的原始思路如下:

(1) 在$\{0,1\}^m$ 空间中随机地取定一个点作为 X 的初始点;

(2) 设在任一时刻 X 的值 $X^{(t)}$ 已算出,先按某种自然方式将 $X^{(t)}$ 的邻域中的下降点点集排成直线队列,若下降点点集非空,则将这个队列的为首的点选作点 X 在 $t+1$ 时刻的值 $X^{(t+1)}$.我们可将这种从 $X^{(t)}$ 到 $X^{(t+1)}$ 的过渡称作下降步骤.若下降点点集为空,则知 $X^{(t)}$ 点已为函数 U 的局部最小值点,于是按某种基于拟人思路的“跳坑”策略将计算从 $X^{(t)}$ 点跳到其邻域中的一个确定的新点 $X^{(t+1)}$.我们可将这种从 $X^{(t)}$ 到 $X^{(t+1)}$ 的过渡称作逃出步骤.

以上思路虽然是本质的,但有一个小漏洞需要补平.即在逃出步骤中,若点 $X^{(t)}$ 经翻转某个变元 x_v 后变成 $X^{(t+1)}$,则很可能从 $X^{(t+1)}$ 的角度来看,老点 $X^{(t)}$ 是 $X^{(t+1)}$ 的邻域中的下降点,于是很有可能从 $X^{(t+1)}$ 出发的下降步骤是再翻转变元 x_v 值而得出 $X^{(t+2)} = X^{(t)}$.即计算走了回头路.

为补平这一漏洞,我们可以设置变元禁集,并将下降变元中的诸变元按实行翻转的优先顺序排成纵队列.这样,我们的寻优搜索的思路框架被发展为如下框图:



以下我们分节描述利用什么样的拟人策略及如何具体完成此框图中的各种计算.这种描述一旦完成,我们对 SAT 问题的求解算法事实上也就清楚地呈现了出来.

5.4 继承策略——点 X 上的势能函数与下降变元集的快速计算

记第 t 时刻的诸变元 $x_1, x_2, \cdots, x_m$ 的取值为 $X^{(t)} \in \{0,1\}^m$,于是第 $t+1$ 时刻的点 $X^{(t+1)}$ 是从 $X^{(t)}$ 出发经过翻转某变元 x_u $(u = 1,2,\cdots, m)$的 01 值而来.

由于对(3.8)式中各 U_i,在其表达式(3.9)中,每个变元至多只出现一次,根据分析学中的 Taylor 公式,我们有如下严格等式:

$$U\left[X^{(t+1)}\right] = U\left[X^{(t)}\right] + \left[x_u^{(t+1)} - x_u^{(t)}\right] \cdot \frac{\partial U\left[X^{(t)}\right]}{\partial x_u}.$$

由于 01 值 $x_u^{(t+1)}$ 是自 01 值 $x_u^{(t)}$ 翻转而来, $x_u^{(t+1)} = 1 - x_u^{(t)}$,于是得出计算总势能 $U\left[x_1, x_2, \cdots, x_m\right]$ 的递推公式

$$U\left[X^{(t+1)}\right] = U\left[X^{(t)}\right] + \left[1 - 2x_u^{(t)}\right] \cdot \frac{\partial U\left[X^{(t)}\right]}{\partial x_u}. \tag{5.2}$$

为了计算出点 $X^{(t)}$ 的诸下降变元,只需计算出诸值 $\frac{\partial U\left[X(t)\right]}{\partial x_v}$, $v = 1, 2, \cdots, m$即可,例如,若 $X^{(t)} = (0,1,0,1)$,

$$\frac{\partial U(X^{(t)})}{\partial x_1}=1,\quad \frac{\partial U(X^{(t)})}{\partial x_2}=1,\quad \frac{\partial U(X^{(t)})}{\partial x_3}=-1,\quad \frac{\partial U(X^{(t)})}{\partial x_4}=-1,$$

则可知 x_1 为上升变元, x_2 为下降变元, x_3 为下降变元, x_4 为上升变元.

由于 $\partial U/\partial x_v$ 对于 x_u 的线性性,类似(5.2)式的导出,可得计算诸 $\partial U/\partial x_v$ 的递推公式如下:

$$\begin{cases} \dfrac{\partial U(X^{(t+1)})}{\partial x_v}=\dfrac{\partial U(X^{(t)})}{\partial x_v}+(1-2x_u^{(t)})\cdot\sum_{i=1}^{l}\dfrac{\partial^2 U_i(X^{(t)})}{\partial x_u\,\partial x_v}, \\ \dfrac{\partial U(X^{(t+1)})}{\partial x_u}=\dfrac{\partial U(X^{(t)})}{\partial x_u}, \end{cases} \tag{5.3}$$

其中 $v\in\{1,2,\cdots,m\}-\{u\}$.

递推式(5.3)使我们在计算点 X 上的总势能函数 U 的诸一阶偏导数时可节约大量时间,因为在新情况下的计算中充分地利用了老情况下已有的讯息.而新老情况的差异是很小的,这体现在(5.3)式中的项 $\dfrac{\partial^2 U_i(X^{(t)})}{\partial x_u\,\partial x_v}$ 对于很多的 (i,u,v) 皆为零.至于总势能函数 U,在有了偏导数值后,利用(5.2)式来计算则更是简单.

对于当今国际学术界公认为最困难的 SAT 问题之一类的随机 3-SAT 问题,即(3.7)式中的 $l=4.25\sim 4.3m$, $k_i+K_{ri}=3$,且这 3 个变元由 $(P_1,P_2,\cdots,P_m)$ 中随机选取,然后对每个变元随机地取非或不取非的情形,我们可以分析一下在新点 $X^{(t+1)}$ 上求诸一阶偏导数的工作量.

由于 CNF 中总共有 $3l=3\times 4.3m$ 个文字出现,所以变元 x_u 在 U 的表达式中出现的次数大约为 $3\times 4.3m/m=3\times 4.3\approx 13$ 次,

$$\underbrace{(x_u\cdot\cdot)+(\cdot\, x_u\,\cdot)+\cdots\cdots+(\cdot\cdot\, x_u)}_{\text{约 13 个括号}}.$$

于是,在 U 的表达式中与 x_u 以乘积关系同时出现在某一个圆括号中的变元 x_v(称作搭界变元)大约少于 $2\times 13=26$ 个.即在每一个新时刻,大约只有少于 26 个的一阶偏导数需要更新,而其他的一阶偏导数则不变.

有趣的是,26 是一个绝对常数,与问题的规模即 CNF 的长度无关.在此离散的空间$\{0,1\}^m$ 中,点 $X^{(t+1)}$ 上的势能与下降变元集的这种快速计算,显然是得益于由(3.9)与(3.8)式定义的连续空间$(-\infty,+\infty)^m$ 中具有和谐性质的连续函数 $U(x_1,x_2,\cdots,x_m)$.

5.5 Bart Selman 的"跳坑策略"及其拟人解释

我们采用 Bart Selman 的策略作为"跳坑策略",设在 t 时刻,计算处于

$\{0,1\}^m$ 中的点 $X^{(t)}$，$U\lceil X^{(t)}\rceil = \sum_{i=1}^{l} U_i\lceil X^{(t)}\rceil > 0$，按(3.9)式，显然对每个 i，有 $U_i\lceil X^{(t)}\rceil = 0$ 或 $U_i\lceil X^{(t)}\rceil = 1$. Selman 的策略是随机地选取一个使 $U_i\lceil X^{(t)}\rceil = 1$的函数 U_i，再在此 U_i 的表达式中随机地选取一个变元，将其 01 值翻转.这样，U_i 的值由 1 变成 0，点 $X^{(t)}$ 变成点 $X^{(t+1)}$.

可从拟人的角度对 Selman 的策略作如下体味.函数 $U\lceil X^{(t)}\rceil$ 的值是整个社会对点 $X^{(t)}$ 的不满意的总程度的度量.整个社会由 l 个集团 U_i ($i = 1,2,\cdots,l$)组成，每个集团对点 $X^{(t)}$ 的不满意的程度为相应的函数值 $U_i\lceil X^{(t)}\rceil$，它要么为 0 要么为 1.Selman 策略的意味是当局实施如下动作：在整个社会中随便选取一个不满意的集团，然后针对此集团修改一项与它有关的政策，使它由不满意到满意.

当局的这种动作与在诸物体的 Packing 问题的求解过程中让因受挤压感到最痛苦的物体从当前的位置跳走至一个随机地确定的新位置的做法，其精神完全是一致的.

5.6 对新路策略的贯彻

我们描述下降变元队列与变元禁集的演化规则以及有关的赦免策略.

下降变元队列的演化规则如下：

(1) 在初始时刻，即在初始指派 $X^{(0)}$ 之下，将全部下降变元按变元序号排列，小序号变元在先，大序号变元在后；

(2) 每翻转一个变元 x_u 后，点 $X^{(t)}$ 变成了 $X^{(t+1)}$，若 x_u 原来不是点 $X^{(t)}$ 的下降变元，翻转后 x_u 成为了点 $X^{(t+1)}$ 的下降变元，则将变元 x_u 加入到 t 时刻的下降变元队列之尾；

(3) 每翻转一个变元 x_u 后，点 $X^{(t)}$ 变成了 $X^{(t+1)}$，若另有变元 x_v，$v \neq u$，x_v 原来不是 $X^{(t)}$ 的下降变元，翻转后 x_v 成为了 $X^{(t+1)}$ 的下降变元，则将变元 x_v 加入到 t 时刻的下降变元队列之首；

(4) 每翻转一个变元 x_u 后，点 $X^{(t)}$ 变成了 $X^{(t+1)}$，将点 $X^{(t)}$ 的下降变元队列中不再是点 $X^{(t+1)}$ 的下降变元的变元删去.

规则(2)是为了防止计算走回头路，将引起走回头路的下降变元 x_u 的翻转优先级别定得最低.规则(3)针对的情况是，在 x_u 翻转的前后 $\partial u / \partial x_v$ 异号，这就表明在新时刻的新地方 $X^{(t+1)}$，令 x_v 的 01 值改变的方向是一个很新的方向，这种改变降低了函数 U 的值却不是走老路.因此，对这种变元 x_v 的翻转优先级别应定得高.

变元禁集的演化规则如下：

(i) 每翻转一个变元 x_u 后,点 $X^{(t)}$ 变成了 $X^{(t+1)}$.若此种翻转是按下降步骤操作的,则置变元禁集为空;

(ii) 每翻转一个变元 x_u 后,点 $X^{(t)}$ 变成了 $X^{(t+1)}$.若此种翻转是按"跳坑策略"操作的(即逃出),则将变元 x_u 加入到老的变元禁集中去;

(iii) 每翻转一个变元 x_u 后,点 $X^{(t)}$ 变成了 $X^{(t+1)}$.若此种翻转是按"跳坑策略"操作的(即逃出),则对于老禁集中的诸变元 x_v ($v \neq u$)作如下处理:若 x_v 属于原下降变元集但不属于新下降变元集,或者 x_v 不属于原下降变元集但属于新下降变元集,则将变元 x_v 从老变元禁集中删去.

规则(i) 适用的情形是计算点 $X^{(t)}$ 处于老的局部极小点的陷阱之外.这时当然无需在 $X^{(t+1)}$ 点处对 U 的下降计算进行限制.

规则(ii) 的意味与下降变元队列演化规则(2)是一致的.即,将 x_u 放入禁集的含义是防止计算走回头路,禁止下一时刻再翻转变元 x_u,将其值又变回去.

规则(iii)是开禁的规则.即对于禁集中的变元 x_v,它是在过去某个历史时刻被禁锢进来的,到了实质性的情况发生了变化的新时刻就不应再禁锢它了.这个实质性的情况就是"x_v 是否属于当时的下降变元集".

对于下降变元队列与变元禁集的演化规则,我们有如下定理:

定理 在任一时刻,所有不在禁集中的下降变元都连续地排在下降队列的前段.

证 在初始时刻禁集为空,定理成立.在将基于下降变元队列的演化规则与禁集演化规则所作的各类动作进行清晰的分类之后,可知本定理所示的性质在这些动作的前后具有继承性.例如,由列队规则(2)和禁集演化规则(ii),禁集中的下降变元总是加在下降变元队列之尾;由列队规则(3)和禁集演化规则(iii),不在禁集中的下降变元总是加在下降变元队列之首. □

按此定理,在下降步骤,所需翻转的变元明确地就是下降变元队列之首,不需再去寻找.

5.7 实验结果

我们用随机 3-SAT 问题的样例对本节所得算法 Solar 进行了实验测试.为便于与著名的美国算法 GSAT + w 相比较,我们取 $l/m = 4.25$.实验数据如表4.1所示,所使用的计算机为 Sunsparc2.其中平均计算时间为对于算出为可满足的诸样例的平均计算时间.从表中可以看出:

(1) 在 $l/m = 4.25$ 处算出可满足的样例数占总样例数的比例明显高于0.5.这说明 Solar 的稳定性远优于 GSAT + w,因为对于 GSAT + w 而言,这个比例大致等于0.5.这也附带地说明,比值 l/m 的难易转变点,客观上应高于4.25.

表 4.1

变元数 m	子句数 l	平均计算时间/s	总样例数	报告已满足的样例数	尚未满足的样例数
500	2125	21	50	42	8
1000	4250	91	20	16	4
2000	8500	766	10	7	3

(2) 即使在解出更多的可满足的样例的基础上(亦即在样例总难度提高了的基础上)来统计计算时间,Solar 的平均计算时间仍然明显少于 GSAT + w.这说明 Solar 的速度远高于 GSAT + w.

根据本节的分析与实验结果,我们希望,对于今后国际学术界更现实有用的高维情形(如高于 10^5 维)而言,m 维 Euclid 空间 R^m 中的连续物理模型可能会发挥出更明显的作用.人们研制出统一的、对当代计算机科学与技术的各部门真实有用的 SAT 软件已经为时不远.

§6. 求解不等圆 Packing 问题的纯粹拟人方法

设计算法的智慧来自哪里? 人类本身可能是一个非常丰富的源泉.包括人们的感觉、思想、行为和经验.通过对人类过去数千年有关 Packing 行为的体会,经过长期的领悟,我们为不等圆 Packing 问题(图 4.6)得出了“穴”的形式化精确概念.

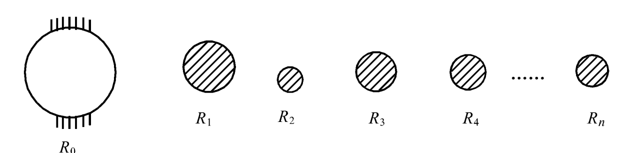


图 4.6　不等圆 Packing 问题

依据占角动作的“最大穴度原则”,我们将外面的圆饼一个接一个地逐步放进容器中去.这已经是一个求解不等圆 Packing 问题的简单的确定型算法,记其为 A_0.

以算法 A_0 为标准和工具再对每一个步骤中的每一个占角动作计算其价值度,然后选择具有最高价值的动作作为当前真实实施的动作.由此 A_0 被发展成为一个新的算法 A_1.

定义 1(占角动作)　一个占角动作是指将外面的某个圆饼放进容器中去,使此圆饼不与容器中已经放好的任何圆饼重叠,并且,它与容器中已经放好的某

两个圆饼相切或者与容器中已经放好的某一个圆饼相切且与容器内壁相切(图 4.7).

定义 2(占角动作的穴度) 直观地表述:动作中被放进的圆饼与容器中的二物体相切.穴度是被用来表征此圆饼与容器内其他物体中离它最近的那个物体的接近程度.如果此圆饼与那个物体的距离为零,则此占角动作的穴度为 1.否则穴度小于 1.穴度为 1 意味着相应的占角动作不仅是占角而且是占了一个完整的“洞穴”.

$$穴度 = 1 - d/R,$$

其中 d 为被放入的圆饼与那个离它最近的另外的物体的距离, R 为被放入的圆饼的半径(图 4.7).

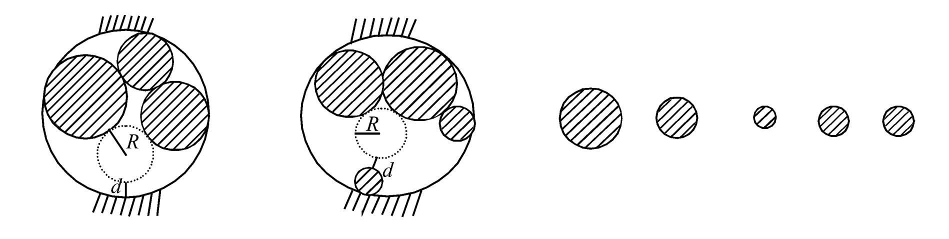


图 4.7 占角动作及其穴度

算法的开局 在初始格局(图 4.6)中,无占角动作可做,更无穴度可言.因此须在圆盘外挑选两个圆饼,放进圆盘中去使这两个圆饼与圆盘之间两两相切(图 4.8).

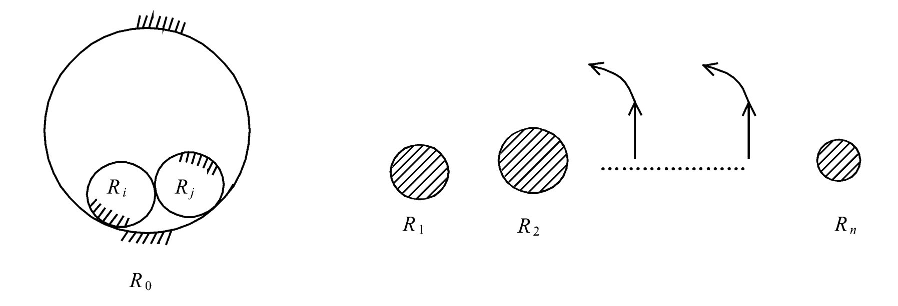
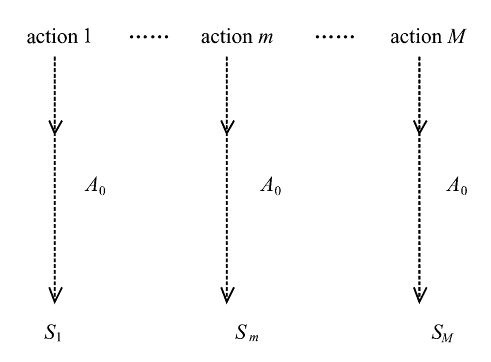


图 4.8 一种开局方式

开局的方式一共有$\frac{1}{2}(n-1)n$ 种.

从 A_0 出发构造 A_1



在开局以后,在 $k(0\leqslant k\leqslant n-2)$ 个放置圆饼的占角动作做完之后,观察圆盘内的 $k+2$ 个位置确定的圆饼及圆盘外的 $n-(k+2)$ 个圆饼.在当前的格局之下试探着做所有可能的占角动作 action 1, …, action M.对其中每一个 action $m(1\leqslant m\leqslant M)$,在做完它之后,继续让算法 A_0 指挥往后的占角动作直至无占角动作可做为止.接着统计在此假想终止时刻已放在圆盘中各圆饼的总面积 S_m.设

$$S_{m^*}=\max\{S_1, S_2, \cdots, S_M\}.$$

然后真实地做占角动作 action m^*.

显然 A_1 是一个求解不等圆 Packing 问题的确定型算法.一个夸张的说法是"利用智能(拟人)枚举方法,我们将算法 A_0 发展成了算法 A_1".

计算实验说明算法 A_1 的性能在目前具有其先进性.

关于历史的注　可以将不等圆 Packing 问题的求解过程,即往圆盘内一个接一个地安放圆饼的过程看成是与上帝下棋.关于围棋,中国古代几千年前就有警语"金角银边草肚皮".对于 Packing 问题我们可将此警语发展为"金角银边草肚皮,价值最高钻石穴",并加以形式化.为这一句话我们断断续续地花去了近 25 年的心力.这也说明了拟人途径的一个不足之处——形成算法的不确定性和艰苦性.

第五章　设计算法与研究计算复杂度的结构的一个工具——有穷损害优先方法

许多做纯粹数学的人不闻不问计算机科学,做计算机科学的人不闻不问纯粹数学,这对学问,对学者是一个很大的损失.例如纯粹数学递归论中的一个十分重要的现代方法——有穷损害优先方法,对于计算机科学中的算法设计就很有价值,很有用处.特别地,对于 NP 难问题求解算法的设计,更是有用.但是计算机科学技术界的人士很少问津它,其中一个重要的原因可能是纯粹数学方面的人士尚未将这个方法整理得十分自然、十分直观、十分好懂,计算机科学技术方面的人士把握起来尚有困难.本章的目的在于将这个方法尽可能叙述得自然、直观、好懂一些,以让计算机科学技术界的人士易于把握.

有穷损害优先方法的精神深深地植根于人类在社会生活中处理人与人之间各种矛盾的公关手腕.例如在争取团结新朋友的时候不要得罪老朋友,实在不得已得罪了,随着往后事态的发展最好能选择合适的时机进行数次成功的安抚,以期最后争取团结到所有的人.又例如在资源紧张时有必要对客户们按价值和影响的大小排一个优先序,以便于有效有序地分配资源.

这个方法是为解决纯粹数学中悬而未决的重要问题——Post 问题而于 1956 年被发明的.发明人为原苏联的数学家 A.A.Muchnik.稍后,美国数学家 R.M.Friedberg 于 1957 年也发明了这个方法.

其实,此方法自发明之日起,在很长一段时期内,连正统的数学家都很难理解、很难把握.其原因可能大致为如下几条:一是这个方法的原始思想是来自于人的社会经验,来自于人们的公关手腕,而不是来自于严格的公理化形式体系中的学问,不是来自于逻辑.近百年来,多数优秀的数学家们已经对这种原始的思想和经验生疏了,渐渐缺乏有关方面的训练、缺乏有关的能力了.第二个原因是方法的发明人 Muchnik 与 Friedberg 当时都十分年轻,才 21 岁.他们的感觉非常丰富,思想十分活跃,在某些新鲜图像的刺激下,在某种灵感的暗示下,他们利用形式证明,仓促地将算法及其正确性肯定了下来以利迅速发表,而还没有来得及对该算法及其原始思想彻底的理解并直观化.第三方面的原因是,当时递归论学术界所拥有的符号和语汇的体系以及表达的习惯相对于有穷损害优先方法来说还不够丰富、方便与贴切.

这种困难局面持续了颇长一段时期,人们只是觉得 Muchnik 和 Friedberg 有道理,相信他们的结果为正确,但是很难说细说透其严格的证明,更难体会把握

其算法的原始思想与直观图像.更有甚者,有一较年长的美国优秀数学家,就是在力图弄清此方法的奋斗中因反复操心、过度疲劳而去世.

后来,A. Nerode 学派的优秀数学家 Robert I. Soare 花了长时期的苦功夫追踪透视有穷损害优先方法,运用这个方法对不可解度的结构进行了深入细致的研究,对这个方法的各个方面进行了严格透彻的证明,并且达到了清晰、亲切、直观.

Robert I. Soare 在 1987 年出版了专著"Recursively Enumerable Sets and Degrees".在此专著中他对有穷损害优先方法给出了真正清楚正确而又明白好懂的描述和证明.

至此,不仅任何一个数学家都能理解这个方法,并且计算机系和数学系三年级的学生都能很好地理解和把握这个方法.

我们稍仔细地分析一下就会发现,计算机科学的核心是求解问题的算法,而算法的核心是化解或处理矛盾.人类在数万年特别是在近几千年的生活中显然积累了大量的处理矛盾的经验,产生了化解矛盾的各种智慧.这些经验与智慧,经过合适的变换和形式化后是能够处理许多困难的算法问题,特别是 NP 难问题的.

两个问题 A 和 B,如果利用问题 A 的答案(犹如对某种函数表的利用)有 Turing 机能在多项式时间里解决问题 B,则说按多项式复杂度问题 A 的复杂度高于等于 B,或说问题 B 的复杂度低于等于 A.如果问题 A 的复杂度高于等于 B 并且 B 的复杂度高于等于 A,则说 A 的复杂度等于 B 的复杂度, B 的复杂度也等于 A 的复杂度.如果问题 A 的复杂度高于等于 B,但是 B 的复杂度不高于等于 A,则说问题 A 的复杂度高于 B 的复杂度,或说问题 B 的复杂度低于 A 的复杂度.

这么一来,按多项式时间复杂度,对任意两个问题就有了复杂度高、低、相等或互相不可比较的概念了.于是对于全体问题就有了依复杂度决定的偏序结构了.研究这一结构当然有其鲜明的理论意义.可以想象,有穷损害优先方法对于这种构造的研究将会是一个很基本的有用工具.

有穷损害优先方法是人们在研究纯粹数学递归论的过程中发明的一个强有力的方法.其原始动机是要证明不可计算度结构中的某些现象.我们感到,其思想和技术可以转移过来设计各种 NP 难度问题的求解算法以及研究计算复杂度的结构.

此方法的实质,对于有点社会经验的普通人都是易于理解的,惟独一些纯正的数学家理解起来觉得十分困难.此方法由原苏联数学家 Muchnik 与美国的数学家 Friedberg 所发明.

我们通过一个简单具体问题的解决来介绍和体会这个方法.

§1. 递归论中的几个基本概念

在第一章 §4 中我们将全体 Turing 机作了编码,得出了 Turing 机序列

$$T_0, T_1, T_2, T_3, \cdots \tag{1.1}$$

这里任一 Turing 机 T_i 都代表了一个从自然数集 N 映射到自然数集 N 的部分函数 φ_i:

$$\varphi_i(x)=\begin{cases}\text{停机时带上 } s_1 \text{ 符号的个数,如果以格局 } \underset{q_1}{\overset{s_0}{\uparrow}}\ \underbrace{s_1 s_1 \cdots s_1}_{x\text{ 个}} \text{ 开始,} \\ \text{最终 } T_i \text{ 会停机;} \\ \text{无定义,如果以格局 } \underset{q_1}{\overset{s_0}{\uparrow}}\ \underbrace{s_1 s_1 \cdots s_1}_{x\text{ 个}} \text{ 开始 } T_i \text{ 永不停机.}\end{cases}$$

因此,序列(1.1)同时也表示了可计算部分函数的序列

$$\varphi_0, \varphi_1, \varphi_2, \varphi_3, \cdots \tag{1.2}$$

事实上,由 Church–Turing 论题知,从 N 到 N 的任何一个可计算部分函数都会在此序列中出现.

顺便说明,全函数是部分函数的特例.从 N 到 N 的任何一个可计算的全函数也都会在此序列中出现.

T_i 就是实现对函数 φ_i 的计算的一种计算方法.

定义 1 $w_i=\{x \mid x\in N \,\&\, \varphi_i(x)\downarrow\}$, $i=0,1,2,\cdots$.

定义 2 $w_{i,s}=\{x \mid x\leqslant s \,\&\, \varphi_i(x)\downarrow \,\&\, \varphi_i(x)\text{的计算步数}\leqslant s\}$.

$x\in w_{i,s}$ 意味着机器 T_i 用不多于 s 步的计算时间即可发现 $\varphi_i(x)\downarrow$. $w_{i,s}$ 不同于 w_i 之点在于 $w_{i,s}$ 是可计算集,而 w_i 对于某些 i,并不是可计算集.

定义 3 说 N 的子集合 A 为递归可枚举集(r.e.集),是指存在一个从 N 到 N 的可计算的全函数 f,使得

$$A=\{f(0), f(1), f(2), \cdots\}.$$

即递归可枚举集的全体元素是能够能行地一步步地枚举得出来的.

命题 1 $(\forall\, i)\ w_i$ 是 r.e.集.

证 由 $w_i=w_{i,1}\cup w_{i,2}\cup w_{i,3}\cup\cdots$,每个 $w_{i,j}$ 为有穷集,结论成立. □

命题 2 对任一 r.e.集 A,存在自然数 i,使得 $A=w_i$.

证　设 $A=\{f(0),f(1),f(2),\cdots\}$，其中 f 为可计算的全函数.对任一自然数 x，寻找"x 在序列 $f(0),f(1),f(2),\cdots$ 中出现的最早时刻 t"，算法为

$x=?\ f(0)$，若是则输出 $t=0$，停机，否则往下；

$x=?\ f(1)$，若是则输出 $t=1$，停机，否则往下；

$x=?\ f(2)$，若是则输出 $t=2$，停机，否则往下；

……

即函数 $t(x)=(\min s)[x=f(s)]$ 是一个有算法计算的部分函数.按 Church-Turing 论题，有 Turing 机实现对此函数的计算.设其号码为 i，于是 $t(x)=\varphi_i(x)$.

显然有

$x\in A\Rightarrow t(x)$有定义$\Rightarrow t(x)$的计算会停机$\Rightarrow\varphi_i(x)\downarrow$，

$x\notin A\Rightarrow t(x)$无定义$\Rightarrow t(x)$的计算永不停机$\Rightarrow\varphi_i(x)\uparrow$，

即 $A=w_i$. □

命题 1 和命题 2 联合在一起说明序列

$$w_0,\ w_1,\ w_2,\cdots \tag{1.3}$$

是对全体递归可枚举集的一个枚举.这也说明了本节的定义 3 与第 2 章 §3 的定义 1 是一致的.

定义 4　集合 $A\subset\mathrm{N}$ 为单纯集，如果(图 5.1)：

$$\begin{cases}\text{(i)}\ A\ \text{为 r.e.集；}\\ \text{(ii)}\ |\overline{A}|\ \text{为}\ \infty\text{；}\\ \text{(iii)}\ \overline{A}\ \text{中不含有任何一个无穷的 r.e.集.}\end{cases} \tag{1.4}$$

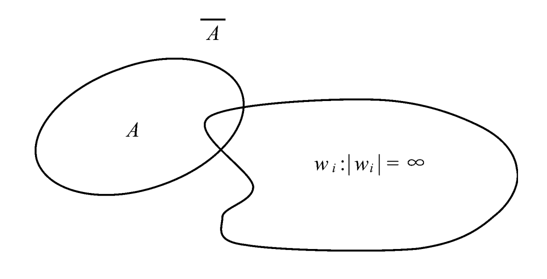


图 5.1

条件(i)是说集 A 是能够能行地被枚举出来的，即能够被人经可数次动作后"选出来"的.条件(ii)是说 A 应充分地小，使得 $\overline{A}$ 充分地大，大到元素个数无

穷,不至于成为平庸集.(iii)是说 A 应充分地大,使得余集 $\overline{A}$ 充分地小,小到装不下任何一个无穷的 r.e.集.

对于集 A 来说,(i)是“可构造性条件”;(ii)是消极条件,即 A 中成员越少越好;(iii)是积极条件,即 A 中成员越多越好.

单纯集若存在,其意义是指,存在着一种“制造方法”,能制造出一个满足两类看起来互相矛盾的条件的集来.

§2. 单纯集的存在性的构造性证明

定理 1 存在着单纯集.

证 只需设计出一种能行的枚举方法,枚举出一个集合 A,使得(ii) $|\overline{A}|$ 为 ∞,(iii) $\overline{A}$ 中不含有任何一个无穷的 w_i.

将条件(ii)“砸开”为(即等价地写为)∞ 个简单的条件.

$$|\overline{A}|\geqslant 0,\ |\overline{A}|\geqslant 1,\cdots,\ |\overline{A}|\geqslant e,\cdots \tag{2.1}$$

将条件(iii)“砸开”为∞ 个简单的条件.

$$|w_0|=\infty\rightarrow w_0\cap A\neq\varnothing,\ |w_1|=\infty\rightarrow w_1\cap A\neq\varnothing,\cdots,$$

$$|w_e|=\infty\rightarrow w_e\cap A\neq\varnothing,\cdots \tag{2.2}$$

将条件 $|\overline{A}|\geqslant e$ 称作第 e 个负要求 N_e,将条件 $|w_e|=\infty\rightarrow w_e\cap A\neq\varnothing$ 称作第 e 个正要求 P_e.

对(2.1)和(2.2)这 $2\cdot\infty$ 个要求可按某种“重要性程度”的标准排个序,将较重要的要求排在前面.往往在具体安排工作中,谁更重要,对谁可马虎些,如何估计这种差别是不重要的事,而对这 $2\cdot\infty$ 个要求排好一个确切的先后序是重要的事情.在我们面临的情形,可采取以下自然的序:

$$N_0,\ P_0,\ N_1,\ P_1,\cdots,\ N_{e-1},\ P_{e-1},\ N_e,\ P_e,\cdots \tag{2.3}$$

为枚举出所要求的集 A,我们用一个堆栈型的计算装置.将此堆栈想象为一个下方有底,上方开口的无穷长的圆柱型容器.就像美国农庄中所用的粮仓.在初始时刻堆栈中装了无穷个银元,贴底放着第 0 号银元,往上顺序放第 1 号,第 2 号,……银元,直至无穷.每个银元厚度为 1,第 i 个银元上刻着数码 i,$i=0,1,2,\cdots$,代表自然数 i.在往后的计算过程中,每一步的动作是确定的,即从堆栈中挖出一个确定的银元扔到旁边地上的大容器中去,或者不挖不扔任何银元.

这样,经过∞ 步之后,地上大容器中的银元集,即自然数子集 A 当然是一个 r.e.集.如果扔银元的计算过程安排得足够地“滑头”,最终照顾了各种要求,则

地上容器中的自然数子集 A 就满足了 $2\cdot\infty$ 个要求中的每一个.即 A 成为单纯集.在计算过程中,若某时刻从堆栈中挖走了一个银元,则上面的无穷个银元全部垮下一格,填满整个堆栈.

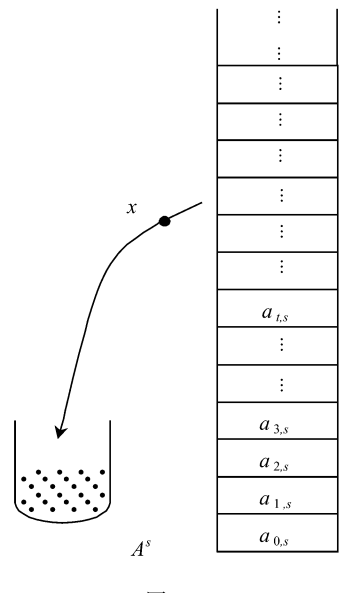


图 5.2

将第 s 步($s=0,1,2,\cdots$)动作之后地上容器中的自然数子集记作 A^s;而将堆栈中的自然数按从下往上的秩序记作

$$a_{0,s},\ a_{1,s},\ a_{2,s},\ \cdots,$$

如图 5.2,最终作出的集为

$$A=\bigcup_{s=0}^{\infty}A^s. \tag{2.4}$$

对 A 的枚举算法

stage　0.令 $A^0=\varnothing$, $\overline{A}^0=\mathrm{N}$.

stage　$s+1$.已知 A^s, $\overline{A}^s=\{a_{0,s},a_{1,s},a_{2,s},\cdots\}$,观察有穷集:$w_{0,s}$, $w_{1,s}$, $w_{2,s}$, $\cdots$, $w_{s,s}$.

找自然数 $t\in[0,s]$,使得

$$\begin{cases} w_{t,s}\cap A^s=\varnothing, & (2.5)\\ (\exists x)[x\in w_{t,s},\ x\geqslant a_{t,s}]. & (2.6)\end{cases}$$

若有此种 t,则找出最小的 t,在此最小 t 之下按(2.6)式找出最小的 x,将此 x 从堆栈中挖出扔入 A^s 中,即令 $A^{s+1}=A^s\cup\{x\}$;

若无此种 t 则不做挖数动作,即 $A^{s+1}=A^s$.

无穷步后所得之集为 $A=\bigcup_{s=0}^{\infty}A^s$.　□

对枚举算法中的策略的说明

一旦从 $w_{t,s}$ 中挖出一数 x 扔入 A^s,即永远地满足了要求 P_t.

"找出最小的 t"意即一旦有机会出现时,我们即做动作去满足最重要的 P_t.此动作做过之后,$x\in A^{s+1}$,于是 $A^{s+1}\cap w_{t,s}\neq\varnothing$,于是 $A\cap w_t\neq\varnothing$,于是 P_t 得以满足.

(2.5)式中"$w_{t,s}\cap A^s=\varnothing$"意即若当前已知要求 P_t 已被满足了,我们即将机会用来满足别的要求,而不去理睬 P_t 这个要求.当前的动作只用来满足当前未被满足的要求.

(2.6)式中的"$x\in w_{t,s}$"意即我们急于要去满足所有的需求 P_t.

(2.6)式中的“$x \geqslant a_{t,s}$”意即为了满足一个正需求 P_t,不能损害更重要的负需求 N_t:$|\overline{A}| \geqslant t$.

“若无此种 t 则不做动作,即 $A^{s+1} = A^s$”意即如果现在没有合适的机会,则甘愿耐心等待,不做任何事情,只是注视事态的发展,直到在新的时刻,出现新的机遇.

“$|\overline{A}| \geqslant 0, |\overline{A}| \geqslant 1, |\overline{A}| \geqslant 2, \cdots$” 意即砸碎一个难满足的要求 $|\overline{A}| = \infty$ 为∞个容易满足的要求,在以后无穷长的工作进程中,我们最终一一满足所有这∞个要求.

对枚举算法的正确性的证明

0. A 为 r.e.集,因为 $A = \bigcup\limits_{s=0}^{\infty} A^s$,而对每一个 s,我们都能算出有穷集 A^s 是由哪些自然数组成.

1. 说需求 N_e 在第 $s+1$ 步被 x 所损害,是指 $x \in A^{s+1} - A^s \wedge x < a_{e,s}$.我们有命题:对任一自然数 e, N_e 至多被损害有穷次.

证 考虑集 $w_0, w_1, \cdots, w_{e-1}, w_e$.在其中最终是取出过数扔进 A 中的集 w_i(例如 $w_0, w_2, w_3, \cdots, w_{e-1}, w_e$)都做了一次扔数的动作之后,即 s 充分大了之后, $w_0, w_1, w_2, w_3, \cdots, w_{e-1}, w_e$,再永不向 A 中扔数了.这是因为(2.5)式限制了任一集 w_t 至多只向 A 中扔一次数.

于是,以后被扔入 A^s 中的数 x 均来自 $w_{e+1}, w_{e+2}, \cdots$ 都 $\geqslant a_{e+1,s}$.即 N_0, $N_1, \cdots, N_e$ 均不受损害. □

2. $|\overline{A}| = \infty$.

证 由1知需求 $N_0, N_1, \cdots, N_e$ 在 s 充分大以后都不再受损害,于是 $|\overline{A}| \geqslant e$.由于 e 可以是任意给定的自然数, $|\overline{A}| = \infty$. □

3. $P_0, P_1, P_2, \cdots$个个都得到了满足.

证 否则,设 P_e 为指标最小的那个未被满足的需求,即 $P_0, P_1, \cdots, P_{e-1}$ 皆被满足了,但 P_e 未被满足.于是

$$|w_e| = \infty \rightarrow w_e \cap A \neq \varnothing$$

为假,即

$$|w_e| = \infty, w_e \cap A = \varnothing. \tag{2.7}$$

于是当 s 充分大,大到 $w_0, w_1, \cdots, w_{e-1}$ 中的集永不再向 A 中扔数了,且 $s > e$ 之后有堆栈中的第0,第1,…,第 e 个数永不再更动.这是因为 $w_0, w_1, \cdots, w_{e-1}$ 再不作贡献,又由反证法,假设 $w_e \cap A = \varnothing$,即 w_e 永不作贡献.作贡献的集只可能是 $w_{e+1}, w_{e+2}, \cdots$.于是对于更充分大的 s,对于 $t = e$ 的 t 满足(2.5)和(2.6)式.

这是因为$\lim_{s\to\infty} w_{t,s} = w_t$,而$|w_t| = |w_e| = \infty$,又 $a_{t,s}$当 s 充分大后不变,(2.6)式成立.另外,由(2.7)式中的 $w_e \cap A = \varnothing$,知(2.5)式成立.

注意此时 $w_0, w_1, \cdots, w_{e-1}$已不向 A 作贡献了,于是 $t = e$ 为$[0, s]$中最小的满足条件(2.5)和(2.6)式的 t.因而,在步 $s+1$ 之末有数 x 自 $w_{e,s}$中被扔入 A 中,即

$$A^{s+1} \cap w_{e,s} \neq \varnothing,$$

于是,$A \cap w_e \neq \varnothing$.矛盾. □

§3. 对有穷损害优先方法的几点评注

(1) 此方法在哲学上有点像用以求非线性多元函数最小值点的 Cauchy 最陡下降法.即在所知的局部环境下,尽可能最大的努力,做出一个动作,使达到最好的收益.然后在新的环境下再做新的动作.方法本质上是局部的.

这种方法在将需求分出等级,以及应该损伤谁补益谁,等待时机坚持不断的观察与工作方面,与人类社会在军事、政治、商业斗争与竞争中所使用的手腕特别一致.这也正是纯正的数学家觉得不好理解的原因.

(2) 考虑一个问题,用此方法解,如果成功了,就加以整理,用严格的证明肯定下来;如果失败了,就无声地予以摒弃.不过,由于尽了可能最大的努力,对所有客观的机会也作了充分的利用,往往能取得成功.

(3) 用有穷损害优先方法解决问题,在正式使用方法之前,有一个转化原始问题成为宜于此方法的新问题的过程.这个过程,与其说是数学,不如说是艺术,需要丰富的经验与感觉.

对于无穷个要求如何给出各自的优先级的问题,即“如何”排序的问题,往往不是很重要,重要的是要“有”一个先后秩序,具体谁先谁后都可以.

参 考 文 献

[1] H.柏格森著.时间与自由意志.吴士栋译.北京:商务印书馆,1958

[2] 陈国良.神经网络用于求解组合优化问题.中国神经网络首届学术会议.特邀论文.1990. C(2)N(2)·90

[3] 陈国良.神经计算及其在组合优化中的应用.计算机研究与发展,1992,29(5):1~ 21

[4] 陈国良.并行遗传算法.扬州师院学报(自然科学版),1995,115(3):1~ 9

[5] 陈国良,韩文廷.人工神经网络理论研究进展,电子学报,1996,24(2):70~ 75

[6] 陈国良,谢幸,徐云,顾钧.随机算法重启策略的构造及其在 TSP 中的应用.计算机学报,2002,25(5):514~ 519

[7] 陈国良,张永民.改进的多层栅格嵌入算法.计算机学报,1991,14(5):332~ 339

[8] 陈志祥,赖楚生,黄文奇.关于求解 DNF 永真性问题的近似快速算法的研究.计算机学报,1990,13(10):779~ 786

[9] S.A.Cook. The complexity of theorem-proving procedures. Poc.3rd ACM Symposium on Theory of Computing. 1971.151~ 158

[10] J.M.Crawford, L.D.Auton. Experimental results on the Crossover point in satisfiability problems, Proc.11th AAAI.1993.21~ 27

[11] 崔国华,洪帆,余祥宣.确定平面点集凸包的一类最优算法.计算机学报,1997,20(4):330~ 334

[12] M.D.Davis,E.J.Weyuker 著.可计算性、复杂性、语言:理论计算机科学基础.张立昂,陈进元,耿素云译.北京:清华大学出版社,1989

[13] Marco Dorigo, Luca Maria Gambardella. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transaction on Evolutionary Computation. 1997, Vol.1, April

[14] Feng Enmin, Wang Xilu, Wang Xiumei, Teng Hongfei. An algorithm of global optimization for solving layout problems. European Journal of Operational Research, 1999, 114: 430~ 436.

[15] 冯玉才,黄文奇,周旋.求解雷达群监视目标群问题的拟物算法.中国科学(A 辑),1995,25(9):982~ 988

[16] X.S.Gao, K.Jiang, C.C.Zhu. Geometric Constraint Solving with Conics and Linkages. Computer Aided Design, 2002, 34(6):421~ 433

[17] X.S.Gao, D.M.Wang, L. Yang (eds). Automated Deduction in Geometry. Berlin: Springer-Verlag, 1999

[18] X.S.Gao, J.Z.Zang, S.C.Chou. Geometry Expert. Taipei: Chiu Chang Mathematics

Publishers,1998

[19] X.S.Gao, C. Zhu, S.C.Chou, J.X.Ge. Automated Generation of Kempe Linkages for Algebraic Curves and Surfaces. Mechanism and Machine Theory, 2002,36(9):1019~1033

[20] M.R.Garey, D.S.Johnson 著.计算机和难解性:NP 完全理论导引.张立昂,沈弘,毕源章译.北京:科学出版社,1987

[21] K.Gödel.Über formal mentscheidbare Sätze der Principia Mathematica and verwandter System I. Monatsh.Math.Phys,1931,38:173~198

[22] J.Gu.Efficient local search for very large-scale satisfiability problems. ACM SIGART Bulletin,1992,3(1):8~21

[23] Gu Qiang, Wu Jing,Zhu Hong. An online algorithm of finding longest periodic Subwords. Computers and Artificial Intelligence,1994,13(1):13~24

[24] 关霭雯.吴文俊消元法讲义.北京:北京理工大学出版社,1994

[25] 郝志峰,邹波涛,陈光中.求解点覆盖问题的拟物转换及算法.运筹学学报,1999,3(1):71~75

[26] 贺思敏.可满足性问题的算法设计与分析.清华大学科学与技术系博士论文.北京,1997

[27] 赫胥黎著.进化论与伦理学.《进化论与伦理学》翻译组译.北京:科学出版社,1971

[28] 赫胥黎著.人类在自然界的位置.《人类在自然界的位置》翻译组译.北京:科学出版社,1971

[29] Hilbert,D. Grundlagen der Geometrie. Seventh edition. Leipzig and Berlin,1930

[30] Hilbert,D.,Bernays,P. Grundlagen der Mathematik,vol.1, Berlin,1934;Vol.2,Berlin,1939

[31] J.H.Holland.Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor,1975

[32] 洪帆,蔡蔚,余祥宣.一个用于多级安全关系数据库系统的改进 Bell-La Padula 模型.计算机学报,1995,18(10):763~769

[33] 洪帆,余祥宣.一个用于授权传递的改进 Bell-La Padula 模型.软件学报,1996,19(2):100~105

[34] J.J.Hopfield et.al. Computing with neural Circuits:A model. Science,1986,233:625~633

[35] 胡劲松,陈国良,郭光灿.在量子计算机上求解 0/1 背包问题.计算机学报,1999,22(12):1314~1316

[36] Huang Wenqi. A quasi-physical method for solving the covering problem—an approach to tackling NP-hard problems. Abstracts of Papers Presented to the American Mathematical Society,April 1988,ISSUE 57,Vol.9,No.3,P.275

[37] 黄文奇.求解 Covering 问题的拟物方法——NP 难度问题的一个处理途径.计算机学报,1989,12(8):610~616

[38] 黄文奇.处理 NP 难度问题的拟物和拟人方法.国际离散数学和算法研讨会议论文集(苏运霖主编).广州:暨南大学出版社,1994. 89~ 91

[39] 黄文奇,陈亮.求解空间利用调度问题的拟物方法.中国科学(A 辑),1991,(3):325~ 331

[40] 黄文奇,金人超.求解 SAT 问题的拟物拟人算法—Solar.中国科学(E 辑),1997,27(2):179~ 186

[41] Huang Wenqi, Kang Yan. A heuristic quasi-physical strategy for solving disks packing problem. Simulation Modelling Practice and Theory, 2002, 10:195~ 207

[42] Huang Wenqi, Li Yu, Sylvain, Li Chumin, Xu Ruchu. A "Learning from human" heuristic for solving unequal circle packing problem. Proc. of the First International Workshop on Heuristics. Beijing. 2002, 4:39~ 45

[43] 黄文奇,宋恩民,陈亮,王权利.对于象棋的不败算法.华中理工大学学报,1995,23(5):1~ 4

[44] Huang Wenqi, Wang Gangqiang. A quasi-mechanical method for solving the rectangle covering problem-an approach to tackling NP hard problems. Graphical Models and Image Processing, 1994, 56(3):267~ 271

[45] 黄文奇,许如初.求解 NP-hard 问题的拟人方法.江西师范大学学报(自然科学版),1998,22(增刊):78

[46] 黄文奇,许如初.支持求解圆形 Packing 问题的两个拟人策略.中国科学(E 辑),1999,29(4):347~ 353

[47] 黄文奇,许如初,陈卫东,张京芬.求解 Packing 及 CNF-SAT 问题的拟物拟人方法.华中理工大学学报,1998,26(9):5~ 7

[48] Huang Wenqi, Yu Xiangdong. A DNF without regular shortest consensus path. SIAM J. Comput., 1987, 16(5):836~ 840

[49] Huang W. Q., Zhan S. H. A quasi-physical method of solving packing problems. Mathematical Reviews, American Mathematical Society, 1982, 82h:52002

[50] 黄文奇,詹叔浩.求解 Packing 问题的拟物方法.应用数学学报,1979,2(2):176~ 180

[51] 黄文奇,赵孝武.求解正交数组问题的拟物拟人算法.计算机研究与发展,2002,39(2):205~ 212

[52] 黄文奇,朱虹,许向阳,宋益民.求解方格 Packing 问题的启发式算法.计算机学报,1993,6(11):829~ 836

[53] 蒋昌俊.一类同步合成网合成法发射序列判定的一个多项式时间算法.中国科学(E 辑),2002,32(1):116~ 124

[54] 金人超,宋恩民,黄文奇.布尔函数线路复杂度的一个新的下界.计算机学报,1994,17(5):376~ 379

[55] Kang Lishan, Li Yuanxiang, Pan Zhenjun, He Jun, D. J. Evans. Massively parallel algorithms from physics and biology. International Journal of Computer Mathematics, 2001, 77:201~ 250

[56] 康立山,孙乐林,陈毓屏.解数学物理问题的异步并行算法.北京:科学出版社,1985
[57] 康立山,谢云,尤矢勇,罗祖华.非数值并行算法(第一册)模拟退火算法.北京:科学出版社,1994
[58] 康雁,黄文奇.求解圆形 Packing 问题的一个启发式算法.计算机研究与发展,2002,39(4):410~414
[59] Kapur D., Saxena T., Yang L. Algebraic and geometric reasoning using Dixon resultants. Proc. ISSAC'94, ACM Press. 1994. 99~107
[60] S.Kirkpatrick, C.D.Gelatt, M.P.Vecchi. Optimization by simulated annealing. Science, 1983,220:671~689
[61] Л.Д.朗道, E.M.栗弗席兹著.力学.莫斯科大学物理系四年级中国留学生译.北京:高等教育出版社,1959
[62] Л.Д.朗道, E.M.栗弗席兹著.场论.任朗,袁炳南译.北京:人民教育出版社,1959
[63] Л.Д.朗道, E.M.栗弗席兹著.连续介质力学.彭旭麟译.北京:人民教育出版社,1958
[64] 老聃著.老子.任继愈译注.上海:上海古籍出版社,1988
[65] H.R.Lewis, C.H.Papadimitriou 著.计算理论基础(第二版).张立昂,刘田译.北京:清华大学出版社,2000
[66] Li Huilin, Li Lian, Liu Yixun. The dicision algorithm for linear sentences on PFM. Annals Pure and Applying Logic,1993,59:273~286
[67] 李廉,李慧陵,刘义循.线性方程组的量词可满足性.中国科学(A 辑),1992,(7)
[68] 李未.实现 Ada 语言任务通讯的一个高效算法.中国科学(A 辑),1984,12:1119~1127
[69] 李未.一个开放的逻辑系统.中国科学(A 辑),1992,10:1103~1113
[70] 李未.形式化方法的局限性.学术报告."863"计划第一届计算机高级人才训练班,1996
[71] 李未,黄文奇.一个求解合取范式可满足性问题的数学物理方法.中国科学(A 辑),1994,11:1208~1217
[72] 李未,王驹.力迫演算在开放逻辑系统中的嵌入.科学通报,1994,39(20):1837~1840
[73] 李未,张玉平.描叙认识进程的抽象逻辑.中国科学(A 辑),1996,26(2):167~178
[74] 李祥.可计算性理论导引.贵州:贵州人民出版社,1986
[75] 李言照,腾弘飞,钟万勰.旋转舱中长方体群的装填布局优化.宇航学报,1993,(1):37~43
[76] Liu Pu, Kang Lishan, Hugo de Garis, Chen Yuping. An asynchronous prallel evolutionary algorithm for solving complex non-linear real world optimization problems. Neural, Parallel & Scientific Computations,2002,10(2):179~188
[77] 刘涛,李国杰. 求解 SAT 问题的分级重排搜索算法.软件学报,1996,7(4):204~209
[78] 刘叙华.基于归结方法的自动推理.北京:科学出版社,1994
[79] 刘勇,康立山,陈毓屏.非数值并行算法(第二册)遗传算法.北京:科学出版社,1995
[80] 马绍汉.算法设计与分析.济南:山东大学出版社,1992
[81] Ma Shaohan, Liang Dongmin. A polynomial-time algorithm for reducing the number of variables in MAX SAT problem. Science in China (Series E),1997,40(3):301~311

[82] 马绍汉,梁东敏.缩减 MAX-SAT 问题中变元个数的多项式时间算法.中国科学(E辑), 1997,27(3):260~ 267

[83] 马绍汉,孙伟,陶雪红.一类命题知识库的更新算法.计算机学报,1996,19(10):741~746

[84] A.Nerode,黄文奇.纯粹递归论在递归分析中的应用.数学学报,1985,28(5):625~ 636

[85] 齐白石.白石老人自述.济南:山东画报出版社,2001

[86] Qu Huiqin,Zhu Hong,Peng Chao. New algorithms for some NP-optimization problems by DNA Computing. Progress in Nature Science, 2002,12(6):474~ 478

[87] B.Selman,H.Levesque and D.Mitchell. A new method for solving hard satisfiability problems. Proc.1992,10th AAAI. 440~ 446

[88] 石纯一.基于解释的机器学习方法.北京:清华大学出版社,1997

[89] 石纯一,黄富宁,王家奆.人工智能原理.北京:清华大学出版社,1993

[90] 石纯一,廖士中.定性推理方法.北京:清华大学出版社,2002

[91] 石赫.机械化数学引论.长沙:湖南教育出版社,1998

[92] 史忠植.高级人工智能.北京:科学出版社,1998

[93] 史忠植.知识工程.北京:清华大学出版社,1988

[94] Robert I. Soare, Recursively Enumerable Sets and Degrees——A study of Computable Functions and Computably Generated Sets. Berlin Heidelberg New York Londion Paris Tokyo:Springer-Verlag,1987

[95] 宋恩民,金人超,黄文奇.关于相对化 P=? NP 问题的注记.数学评论与研究,1993,13(3):443~ 450

[96] 孙武著,孙子兵法今译.唐满先译注.南昌:江西人民出版社,1988

[97] A.塔尔斯基,J.C.C.麦克铿赛著.初等代数和几何的判定法.陆钟万译.北京:科学出版社,1959

[98] 腾弘飞,孙守林,葛文海,钟万勰.转动圆桌平衡摆盘——带平衡性能约束的 Packing 问题.中国科学(A辑),1994,24(7):754~ 759

[99] A.M.Turing. On computable numbers with an application to the Entscheidungs problem. Proc.London Math.Soc.,1936,42:230~ 265

[100] 万颖瑜,周智,陈国良,顾钧.Size Scale:求解旅行商问题(TSP)的新算法.计算机研究与发展,2002,39(10):1294~ 1302

[101] Wang Hao. Computation,Logic,Philosophy. Beijing Dodrecht Boston Lansaster Tokyo:Kluwer Academic Publishers/Science Press,1990

[102] Wang Hao. Popular Lectures on Mathematical Logic.Van Nostrand Reinhold Company, Science Press,1981

[103] Wang Hao. Reflections on Kurt Gödel.Massachusetts:The MIT Press,1988

[104] 王能超.同步并行算法设计的二分技术.中国科学(A辑),1995,25(2):207~ 211

[105] 王能超.同步并行算法设计. 北京:科学出版社,1996

[106] Wu Tianjiao. Some test problems on applications of Wu's method in nonlinear program-

ming problems. MM-Res, Preprints, 1991, (6):144~ 155

[107] Wu Tianjiao. On a collision problem. MM-Res, Preprints, 1992, (7):96~ 104

[108] 吴文俊.力学在几何中的一些应用.北京:中国青年出版社,1962

[109] Wu Wentsun. On the decision problem and the mechanization of theorem-proving in elementary geometry. Scientia Sinica, 1978, (21): 159~ 172; re-published in Automated Theorem Proving: after 25 Years(Ed. W. W. Bledsoe & D. W. Loveleand), 1984, 235~ 242

[110] Wu Wentsun. Toward mechanization of geometry——some Comments on Hilbert′s “Grundlagen der Geometrie”. Acta Math, Scientia, 1982, (2): 125~ 138

[111] Wu Wentsun. Some remarks on mechanical theorem -proving in elementary geometry. Act Math. Scientia, 1983, (3): 357~ 360

[112] Wu Wentsun. Basic Principles of Mechanical Theorem Proving in Geometries(Part on Elementary Geometries), (in Chinese). Beijing: Science Press, 1984. English translation by D. M. Wang et al, Springer, 1994

[113] Wu Wentsun. Basic principles of mechanical theorem-proving in elementary geometries. J. Sys. Sci. & Math. Scis., 1984, (4):207~ 235. Re-published in J. of Automated Reasoning, 1986, (2):221~ 252

[114] Wu Wentsun. A survey of developments of mathematics mechanization in China. Chinese Mathematics into 21st century(Eds. W. T. Wu & M. D. Cheng). Beijing: Peking University Press, 1991. 15~ 40

[115] Wu Wentsun. Geometry problem-solving and its Comtemporary Significance. in Proc. First Asian Tech. Conf. in Math., Asso of Math. Educators, Singapore. 1995. 67~ 72

[116] Wu Wentsun. Mathematics Mechanization, Mechanical Geometry Theorem-Proving, Mechanical Geometry Problem-Solving, and Polynomial Equations-Solving. Beijing Dodrecht Boston Lancaster Tokyo: Science Press/Kluwer Academic Publishers, 2000

[117] 吴文俊,张景中,刘卓军,杨路,侯晓荣.王者之路.长沙:湖南科技出版社,1999

[118] Wu Yu-Liang, Huang Wenqi, Lau Siu-chung, C. K. Wong, Gilbert H. Young. An effective quasi-human based heuristic for solving the rectangle packing problem. European Journal of Operational Research, 2002, 141:341~ 358

[119] 谢幸,周智,陈国良,顾钧. 随机竞争策略在 Monte Carlo 算法中的性能分析.计算机学报,2000,23(10):1015~ 1020

[120] 休姆著.人类理解研究.关文运译.北京:商务印书馆,1957

[121] 徐云,陈国良.一种求解难 SAT 问题的改进 DP 算法.中国科学技术大学学报,2002,32(3):358~ 362

[122] 徐云,陈国良,许胤龙,顾钧.O(m^2)时间求解 SAT 问题的随机算法.计算机学报,2001,24(11):1136~ 1141

[123] 许可,李未.随机 K-SAT 问题的回溯算法分析.计算机学报,1996,23(5):452~ 457

[124] 许可,李未.SAT 问题的相变现象.中国科学(E 辑),1999,29(4):354~ 360

[125] 许如初,黄文奇.解不等圆 Packing 问题拟物拟人算法初态选取.武汉:华中理工大学学报,1998,26(4):1~ 3

[126] 许如初,宋恩民,陈卫东.寻求线性规划问题初始基可行解的一种新算法.华中理工大学学报,1997,25(1):105~ 107

[127] Yang L. Automatically solving semi-algebraic systems. Proceedings of ATCM 2001, ATCM, Inc. ,2001.(大会特邀报告)

[128] Yang L. Recent advances on determining the number of real roots of parametric polynomials. Journal of Symbolic Computiation,1999,(28):225~ 242

[129] Yang L. Recent advances in automated theorem proving on inequalities. J. Comput. Sci. & Technol.,1999,14(5):434~ 446

[130] 杨路. 不等式机器证明的降维算法与通用程序. 高技术通讯,1998,8(7):20~ 25

[131] Yang L,Gao X S, Chou S C, Zhang J Z. Automated production of readable proofs for theorems in non-Euclidean geometries. in: Automated Deduction in Geometry, Lecture Notes in Artificial Intelligence 1360,Springer-Verlag,1997.171~ 188

[132] Yang Lu,Hou Xiaorong. Gather-and-Sift: a symbolic method for solving polynomial systems,Proc.ATCM'95(亚洲数学技术科学大会),Singapore.1995.771~ 780

[133] Yang L, Hou X R,Xia B C. A complete algorithm for automated discovering of a class of inequality type theorems. Science in China, Ser. F,2001,44:33~ 49

[134] Yang L, Hou X R, Xia B C. Automated Discovering and Proving for Geometric Inequalities.in: Automated Deduction in Geometry, Lecture Notes in Artificial Intelligence 1669. Springer-Verlag,1999. 30~ 46

[135] Yang L,Hou X R,Zeng Z B. A complete discrimination system for polynomials. Science in China, Ser. E, 1996,39(6):628~ 646

[136] Yang L,Xia B C. An explicit criterion to determine the number of roots in an interval of a polynomial.Progress in Natural Science,2000,10(12):897~ 910

[137] Yang L,Zhang J Z. Searching dependency between algebraic equations:an algorithm applied to automated reasoning. in: Artificial intelligence in Mathematics. Oxford University Press. 1994. 147~ 156

[138] 杨路,张景中,侯晓荣. 非线性代数方程组与定理机器证明.上海:上海科技教育出版社,1996

[139] Yang Lu,Zhang Ju. A practical program of automated proving for a class of geometric inequalities. Automated Deduction in Geometry, Lecture Notes in Artificial Intelligence 2061. Springer - Verlag,2001.41~ 57

[140] 姚新,陈国良.模拟退火算法及其应用.计算机研究与发展.1990,27(7):1~ 6

[141] 姚新,陈国良.神经计算机.计算机工程与应用,1990,(8):44~ 59

[142] 姚新,陈国良,徐惠敏,刘勇.进化算法研究进展.计算机学报,1995,18(9):694~ 706

[143] 佚名著.三十六计.李炳彦译注.北京:解放军出版社,1989

[144] 余祥宣,崔国华,邹海明.计算机算法基础(第二版).武汉:华中理工大学出版社,2000

[145] Zhang Bo, Zhang Ling. The Comparison between the statistical hearistic search and A* . J.of Comput Sci & Technol, 1989,4(2):126~ 132

[146] Zhang Bo, Zhang Ling. Hierarchy and statistical heuristic search. Future Generation Computer Systems (Netherlands), 1990,6(1):43~ 47

[147] Zhang Bo, Zhang Ling. Theory and Applications of Problem Solving. North-Holland: Elsevier Science Publishers B.V.,1992

[148] 张德富,黄文奇,汪厚祥.求解 SAT 问题的拟人退火算法.计算机学报,2002,25(2):148~ 152

[149] 张端明.世纪之交的物理学.武汉:湖北教育出版社,1999

[150] 张端明,李智华,郁伯铭等.脉冲激光沉积制膜的动力学模拟.中国科学(A 辑),2001,31(8):743~ 753

[151] 张健.逻辑公式的可满足性判定——方法、工具及应用.北京:科学出版社,2000

[152] J. Zhang. Constructing Finite Algebras with FALCON. J. of Automated Reasoning, 1996,17(1):1~ 22

[153] 张健.模态逻辑推理的翻译方法.计算机研究与发展,1998,35(5):389~ 392

[154] J.Zhang. System description: MCS: Model-based conjecture searching. Proc.CADE-16. LNAI 1632.1999. 393~ 397

[155] 张景中,杨路,侯晓荣.几何定理机器证明的结式矩阵法.系统科学与数学,1995,15(1):10~ 15,

[156] 张健.有限构模器的扩展及其在形式化方法中的应用.计算机学报.2000,23(2):190~ 194

[157] J. Zhang, H. Zhang. Combining local search and backtracking techniques for constraint satisfaction. Proc.13th AAAI. 1996,1:369~ 374

[158] 张景中.计算机怎样解几何题.北京:清华大学出版社,2000

[159] 张景中.几何问题的机器求解.科学,2001,53(2):20~ 23

[160] 张景中,高小山,周咸青.基于前推法的几何信息搜索系统.计算机学报,1996,19(10):721~ 727

[161] Zhang Jingzhong, Yang Lu, Deng Mike. The parallel numerical method of mechanical theorem proving. Theoretical Computer Science, 1990,74:253~ 271

[162] 张伟,石纯一.Agent 组织的一种递归模型.软件学报,2002,13(11):2149~ 2154

[163] 张立昂.可计算性与计算复杂性导引.北京:北京大学出版社,1996

[164] Zhang Liang. The Complexity of Approximation for K-KNAPSACK.国际离散数学与算法研讨会文集.广州:暨南大学出版社,1994

[165] 张铃,张钹.PLN 网络吸引区域的定量分析.软件学报,1994,5(8):9~ 13

[166] 张铃,张钹.遗传算法机理研究.软件学报,2001,11(7):945~ 952

[167] 张铃,张钹.人工神经网络理论及应用.杭州:浙江科技出版社,1997

[168] 张群,宋中山. Application of machine Computation for studying graphlike manifolds.数学季刊,1997,9(3):101~ 110

[169] 赵孝武,黄文奇.关于SAT问题物理模型猜想的一个反例.华中理工大学学报,2000,28(11):25~ 27

[170] Zhou Xianqing,Gao Xiaoshan,Zhang Jingzhong. Machine Proofs in Geometry.Singapore:World Scientific,1994

[171] Zhao Yunlei,Zhu Hong. Efficient 4-round zero-knowledge proof system for NP. Progress in Natural Science,2002,12(12):948~ 952

[172] 朱大铭,马绍汉.二进制神经网络分类问题的几何学习算法.软件学报,1997,8(8):622~ 629

[173] 朱大铭,马绍汉.基因组 Translocation 排序问题的改进多项式算法.计算机学报,2002,125(2):189~ 196

[174] 朱洪.一个有效的稳定并队算法.计算机学报,1985,8(5):369~ 380

[175] 朱洪.知识和复杂性.知识科学与计算科学(陆汝钤主编).北京:清华大学出版社 2003.35~ 48

[176] 朱洪,鲍振东,李为鉴.关于安全质数的若干特性.通信保密,1987,(3):1~ 6

[177] Zhu Hong,Li Ke. The Complexity of Transformation From Cyclic to Acyclic Database Schemes. Computers and Artificial Intelligence, 1987,6(5):441~ 447

[178] 朱洪,卢先捷. Richard Denis.算术结构的不可判定性.数学年刊,1997,18A(6):667~672

[179] 邹波涛,郝志峰,陈光中.拟物方法在优化问题中的应用.多目标决策进展98.Hong Kong:Global-Link Publishing Co..193~ 199